

MAGAZINE MSX
AÑO III
Núm. 20
Enero 1987
300 Ptas.
Sprites: un programa para MSX II
El generador de sonido del SVI 318-328
Código máquina: operaciones aritméticas
Basic: introducción de datos
GML: EL LENGUAJE GRAFICO

DIRECTOR:
Juan Arencibia.
COLABORADORES:
Angel Zarazaga, Teresa Aranda, Ricardo García.
DISEÑO:
Benito Gil.
Editada por:
PUBLINFORMATICA, S.A.
C/ Bravo Murillo, 377 - 5.° A
Tel.: 733 71 13
28020 Madrid.
Telex 488877 OPZXE
PRESIDENTE:
Fernando Bolín.
DIRECTOR EDITORIAL REVISTAS DE USUARIOS:
Juan Arencibia.
DIRECTOR DE VENTAS:
Antonio González.
JEFE DE PRODUCCION:
Miguel Onieva.
SERVICIO AL CLIENTE:
Julia González.
Tel.: 733 79 69
DIRECCION, REDACCION Y ADMINISTRACION:
C/ Bravo Murillo, 377 - 5.° A
Tel.: 733 74 13
28020 Madrid.
COORDINADORA DE PUBLICIDAD:
Silvia Bolín.
PUUBLICIDAD EN MADRID:
Emilio García.
PUBLICIDAD EN BARCELONA:
Lidia Cendros.
C/ Pelayo, 12.
Tel.: (93) 301 47 00 Ext. 27-28
08001 Barcelona.
Depósito Legal: M. 16.755-1985
Impreso en G. Velasco, S.A.
C/ Antonio Cabezón, 13. Madrid.
Distribuye:
S.G.E.L. Avda. Valdelaparra, s/n. Alcobendas (Madrid).
DISTRIBUIDORES:
VENEZUELA: SIPAM, S.A. Avda. República Dominicana, 541
ARGENTINA: DISTRIBUIDORA INTERCONTINENTAL BUENOS AIRES.

El P.V.P. para Ceuta, Melilla y Canarias, incluido servicio aéreo será de 300 ptas. sin I.V.A.

SUSCRIPCIONES:
Rogamos dirija toda la correspondencia relacionada con suscripciones a:
MSX
EDISA: Tel, 415 97 12
C/López de Hoyos, 141-5.°
28002 MADRID
(Para todos los pagos reseñar solamente MSX)
Para la compra de ejemplares atrasados diríjanse a la propia editorial
MSX
C/Bravo Murillo, 377-5.° A
Tel. 733 74 13 28020 MADRID

Si deseas colaborar en MSX remite tus artículos o programas a Bravo Murillo 377, 5.° A. 28020 Madrid. Los programas deberán estar grabados en cassette y los artículos mecanografiados.
A efectos de remuneración, se analiza cada colaboración aisladamente, estudiando su complejidad y calidad.



Pasadas las Navidades y las distintas ferias habidas durante estas fechas, hemos podido comprobar el lento pero irresistible ascenso del estándar MSX.

Nuevas aplicaciones, ordenadores más potentes y completos, programas de gestión dignos de un PC y un sin fin de novedades han sido las notas más características de este final de año 1986. No cabe la menor duda que el año finalizado, ha marcado una pauta y una tendencia que era inevitable. Por un lado, la bajada de precios al aparecer nuevos ordenadores, ha sido una de las notas más importante para todos los usuarios Por otro lado, el nuevo giro que algunos fabricantes han dado a su política de mercado, como el desarrollo y comercialización de compatibles PC, pone de manifiesto lo que podemos esperar de ello en un futuro aún algo lejano.

Sin embargo, mientras unos se dedican con verdaderos esfuerzos a hacerse con una parte del pastel de los compatibles (al fin y al cabo, un PC es un PC, y si es de IBM mejor que mejor), los fabricantes de MSX continúan aportando importantes novedades al estándar. De hecho, uno de los programas más comercializados de IBM, el dBASE II, ya tiene versión para MSX (ver crítica en la página 29). Con ello, se abre una puerta hasta la fecha cerrada, la de los programas de aplicación para la pequeña empresa.

Efectivamente, con programas como ese y con un ordenador MSX de la II generación (debido a que incorporan la unidad de disco de 3,5 pulgadas, algunos con 720K), se puede empezar a pensar en controlar un pequeño negocio, sin que por ello se pierda imagen, algo que se suele pensar con demasiada ligereza en cuanto vemos a alguien trabajar con un ordenador doméstico. Pero esto es sólo una de las novedades más destacadas, ya que si nos vamos a Holanda o Alemania, países en donde MSX tiene un mercado muy importante, nos encontraremos con que ya existe disco duro Winchester para MSX. Por otro lado, en Japón, Toshiba comercializa un CD ROM de reducidísimas dimensiones, también para MSX, lo que viene a reflejar el imparable ascenso que el estándar está teniendo.

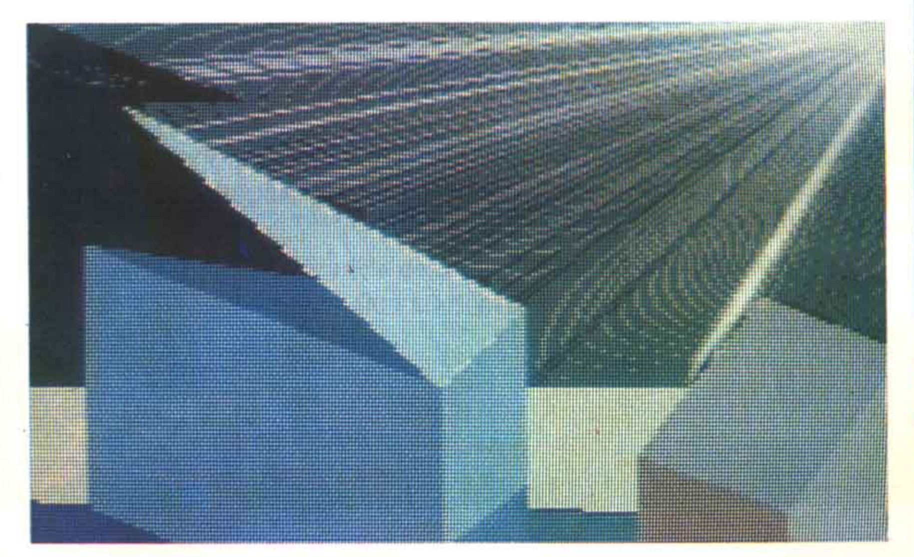
¡¡Hasta el mes que viene!!







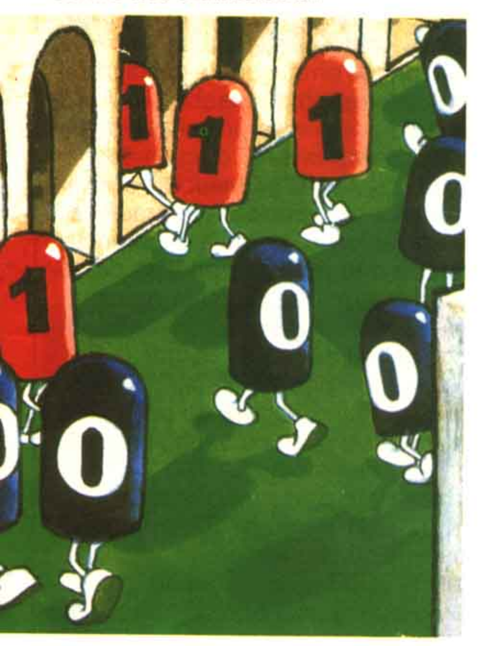
Su






mario





NOTICIAS

Nuevo Inves Spectrum +

Investrónica presenta el nuevo Inves Spectrum +. Este ordenador ha sido desarrollado íntegramente por Investrónica, empresa líder en el desarrollo y lanzamiento de productos basados en la aplicación práctica de Alta Tecnología.

El nuevo Inves Spectrum +, es el ordenador que posee una de las mejores relaciones calidad/precio del mercado. Incorpora una placa totalmente rediseñada, con la que se logran unas notables mejoras.

Otra importante novedad en el Inves Spectrum + es la salida directa para joystick, sin necesidad de acol-

VNU absorbe Hayden

VNU, el más importante grupo editorial holandés, que participa con un 50 por ciento en el capital social de Publinformática, ha suscrito un principio de acuerdo para adquirir la norteamericana *Hayden Publishing*.

Hayden edita, entre otras publicaciones, *Electronic Design, Microwaves & RF, Computer Decisions* y *Personal Computing,* así como la guía directorio *Gold Book. Hayden* está considerada como una de las más importantes editoriales en los campos de la electrónica y la informática, con unos ingresos de 60 millones de dólares en 1986. La absorción, que incluye 290 empleados, será un hecho antes de fin de año.

Así, *Hayden* pasa a formar parte de *VNU Business Press Group,* ya poseedora de destacadas revistas informáticas en el Reino Unido, Holanda y Bélgica, y partícipe en grupos editoriales españoles, franceses, australianos y asiáticos. Tras la adquisición de *Hayden,* la división de Prensa Especializada del grupo VNU contará con unos ingresos anuales de 150 millones de dólares.

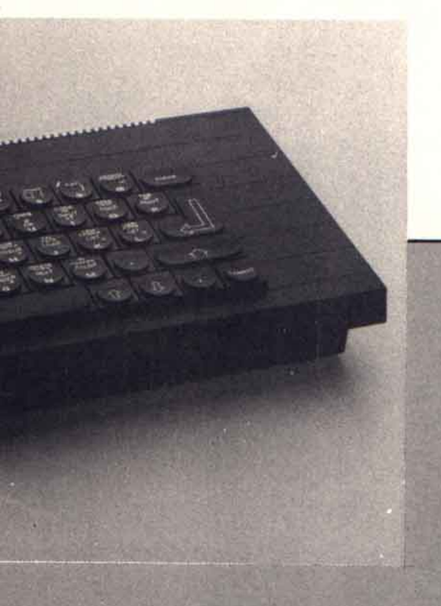
Electronic Design es una revista quincenal especializada en electrónica, con una circulación de 142.000 ejemplares, de los que el 16 por ciento se venden fuera de Estados Unidos. Es la más importante del mundo en su género y está clasificada en séptimo lugar en USA, en cuanto a ingresos por publicidad.

Microwaves & RF, dirigida a ingenieros y jefes de proyecto, se encuadra en el mercado de las radiocomunicaciones de alta frecuencia, tanto en la industria privada como en el área militar. Su venta, mensual, es de 50.000 ejemplares y es la publicación líder en su campo.

El *Gold Book,* por su parte, es una guía-directorio diseñada por y para ingenieros con una venta anual de 120.000 ejemplares.

A su vez, *Computer Decisions* está dirigida a informáticos profesionales y su circulación quincenal es de 175.000 ejemplares, lo cual le proporciona una posición relevante en un sector muy atractivo del mercado.

Finalmente, *Personal Computing* tiene 500.000 ejemplares de tirada, de los que el 90 por ciento corresponde a suscripciones. Esta revista se centra en aplicaciones de gestión para ordenadores personales.



par ningún interface. Es totalmente compatible con todo el software y periféricos de Spectrum y Spectrum +, con más de 15.000 títulos de software, entre los que se pueden encontrar, programas:

- Educativos.
- Utilidades.
- Juegos clásicos.
- Juegos de Destreza.

Al Inves Spectrum +, se le pueden conectar los periféricos más diversos, como: impresora, lápiz óptico, data tablet, ratón, sintetizador de voz, modems y acopladores acústicos, etc.

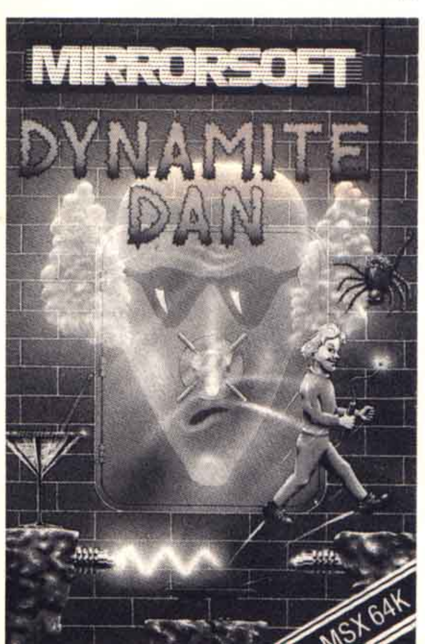
El Inves Spectrum + se entretga con manual de instrucciones en castellano, explicando detalladamente cómo sacar el máximo partido de todas sus características.

Desde Londres con afecto

Así parece ser. Ya que desde esas tierras lejanas nos envía, la empresa Mirrorsoft, dos programas para MSX. *Spitfire 40* y *Dynamite Dan,* son los nuevos títulos que se unirán a la creciente ola de programas para MSX. Ambos vienen en cassette, y con una buena presentación.

Es importante este hecho, puesto que muesta el lento avance del estándar en aquél país, coto particular de caza para Amstrad y Spectrum, enemigos muy importantes con los que se enfrenta en esas tierras.

De *Dynamite Dan,* poco se puede decir. Es un programa que ha escalado hasta las primeras posiciones (versión Spectrum) de las listas de super-ventas. Sin embargo, *Spitfire* 40, es un simulador de vuelo con diversas opciones y sólo apto para frustrados pilotos de combate.





De nuevo Anaya

No podían faltar las novedades de Anaya, empresa que parece no tener techo. En este fin de año, se han comercializado nuevos productos, los cuales vienen a engrosar la amplia biblioteca de libros dedicados a aplicaciones con el ordenador.

«Cálculo Numérico» es una obra dedicada a aquellos lectores inquietos que deseen aprender el Cálculo Numérico, y desarrollar aplicaciones en BASIC para tal fin. Está orientado hacia las necesidades de una amplia gama de ingenieros, estudiantes y científicos que quieren resolver problemas prácticos mediante un ordenador.

Por otro lado, «El Libro del MS-DOS», trata en profundidad al MS-DOS, el sistema operativo del IBM PC (tambie del MSX). Ha sido escrito por programadores de Microsoft, empresa que desarrolló el conocido sistema operativo. Estructurado como un manual de aprendizaje y referencia, está dividido en tres partes: introducción y ejemplos de uso del sistema operativo; tratamiento completo sobre el manejo del MS-DOS y sección de referencia.

Por último, «El IBM PC a fondo», es una obra en la cual se recogen las técnicas más depuradas de trabajo con el IBM PC hasta el momento, y hace una descripción del funcionamiento interno del msimo, centrándose particularmente en el hardware de la máquina más que en su programación.

En conjunto, forman un bloque muy importante e interesante, necesario en cualquier biblioteca técnica.

Lenguaje Gráfico GML

El sistema MSX dispone de una potente herramienta para realizar dibujos libremente, es decir, sin indicación previa del tipo o figura deseados.

Como es natural la mezcla de GML con el resto de instrucciones ya definidas como *LINE, CIRCLE,, PAINT*, etc., proporcionará geométricamente estadística, gestión e incluso arte, un conjunto muy eficiente.

Veremos pues, un resumen de todas las instrucciones generales en primer lugar y después concluiremos con un amplio desarrollo del GML.

Sabemos que *SCREEN* significa pantalla y lo empleamos así:

Ver figura 1

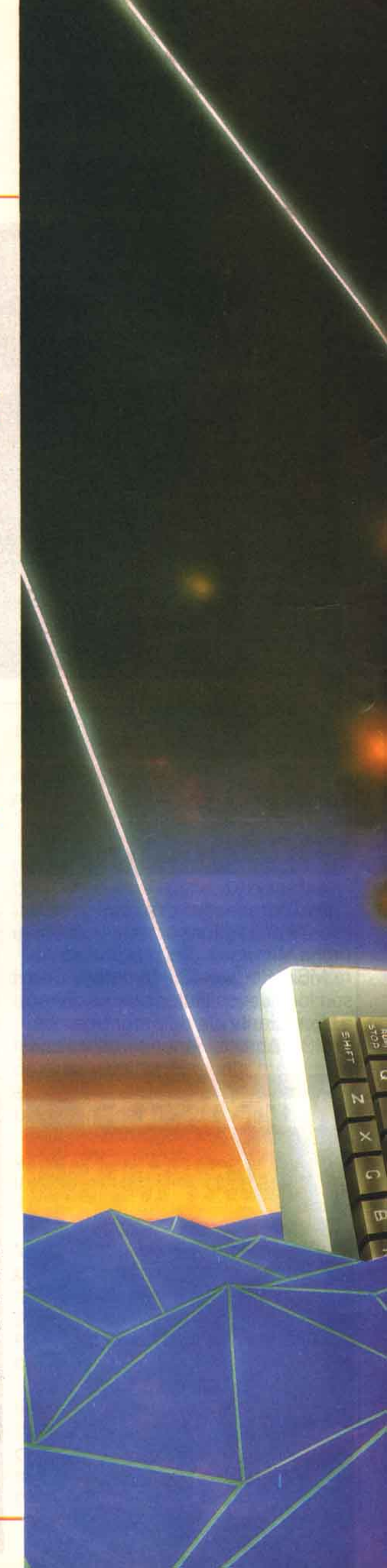
La unidad gráfica es el *pixel* presentado en cualquier de las 49.152 coordenadas correspondientes a la malla que forman 256 filas (x) y 192 columnas (y) a color elegido entre los 16 de que disponemos. Por lo tanto un *pixel* viene determinado por su situación en la pantalla y su color. (Horizontal, Vertical y Color).

Esto constituye la alta definición y comporta:

X (desde 0 hasta 255)
Y (desde 0 hasta 191)

una sola de las dos sería incompleta para la «inteligencia del ordenador; pero el color si no se indica se toma el que esté usándose en ese momento.

Como veremos a base de *pixel* a *pixel* se podrían formar toda clase de figuras: libres, geométricas, mallas, cuadrículas pero demostraremos que son muy laboriosas,





lentas en el desarrollo y por ello poco eficaces.

Para situar un *pixel* en la coordenada deseada debemos indicar la pantalla, específica de la definición que corresponda para que manifieste un punto:

Ver figura 2

Para un color diferente del actual bastará colocar una coma después del paréntesis derecho y a continuación el número que corresponda (0/15).

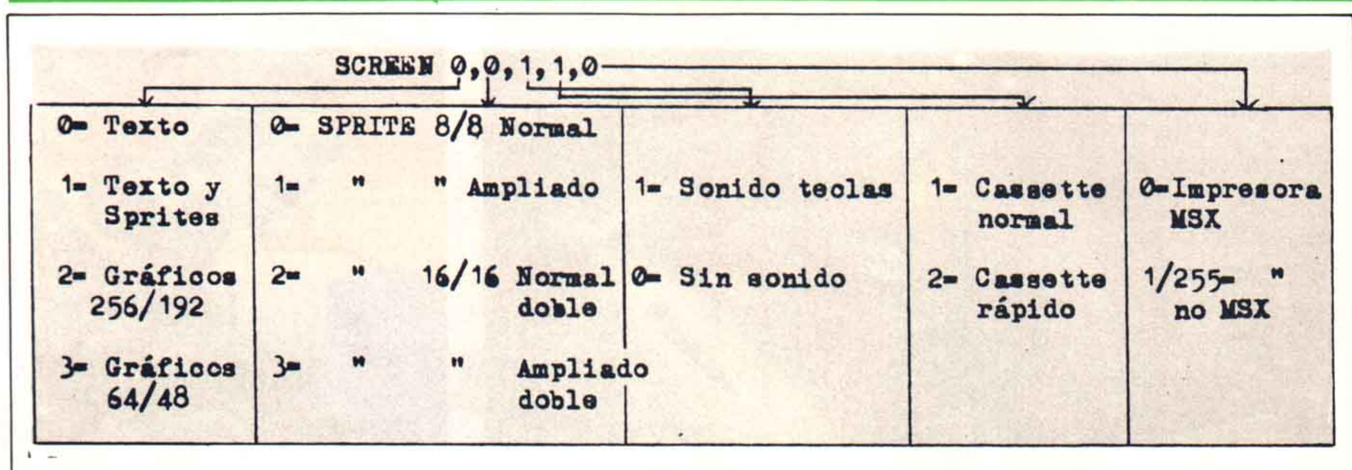
LAS PANTALLAS GRAFICAS NO SON PERMANENTES COMO LAS DE TEXTO, POR ELLO DEBEMOS EMPLEAR UN MEDIO DE FIJARLAS TEMPORAL O PERMANENTEMENTE.

En este caso lo hacemos con un bucle de tiempo 5000. Como los bordes de la pantalla varían entre monitores y aparatos de TV y en estos últimos en casi cada marca hay que ajustarlos como más convenga. Para mayor seguridad dejaremos 10 *pixels* por borde, quedando:

Desde 10 hasta 245 para X
Desde 10 hasta 180 para Y

Ver figura 3

10 Pantalla (SVI la 1).
20 Bucle de apertura para ordenada y.



SCREEN 0,0,1,1,0

0- Texto	0- SPRITE 8/8 Normal			
1- Texto y Sprites	1- " " Ampliado	1- Sonido teclas	1- Cassette normal	0-Impresora MSX
2- Gráficos 256/192	2- " 16/16 Normal doble	0- Sin sonido	2- Cassette rápido	1/255- " no MSX
3- Gráficos 64/48	3- " " Ampliado doble			

Figura 1

30 Idem para la abscisa x con pasos de 10 en 10. Se puede variar el salto a 4 - 20 - 40 - 80... Probad varias veces con distintos números.

40 Clave para el dibujo. Como es punto a punto su configuración se realiza muy despacio. Ya se verán otros sistemas más rápidos.

50 Cierre de ambos bucles (20 y 30).

60 Bucle de apertura abscisa X.

70 Bucle de apertura ordenada Y.

80 Como 40.

90 Como 50 (relativas a 60 y 70).

110 Tope. Para salir emplear CONTROL/STOP.



El MSX «recuerda la última ordenada y abscisa». En el caso de no indicar origen se toma X=0, Y=0 al margen superior izquierda de la pantalla.

La posibilidad de emplear un proceso punto a punto por comando directo es, más bien didáctica, pero interesante para trabajos de programación y su síntesis es:

P S E T (X, Y), C

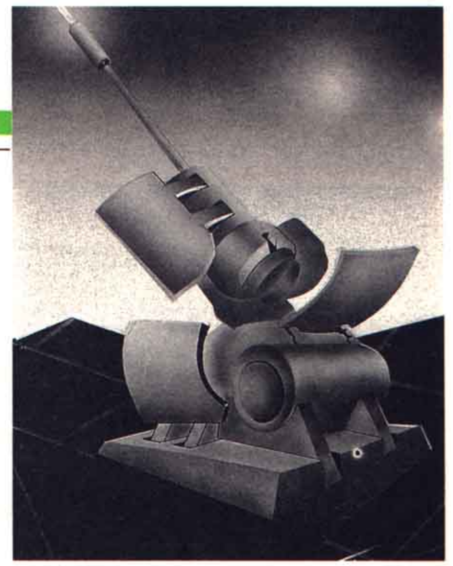
algoritmo en el que X e Y forman la coordenada de cualquier punto dentro de los límites establecidos en columnas/filas y C el color entre los existentes en MSX. (Este último de no especificarse se emplea automáticamente el de primer plano).

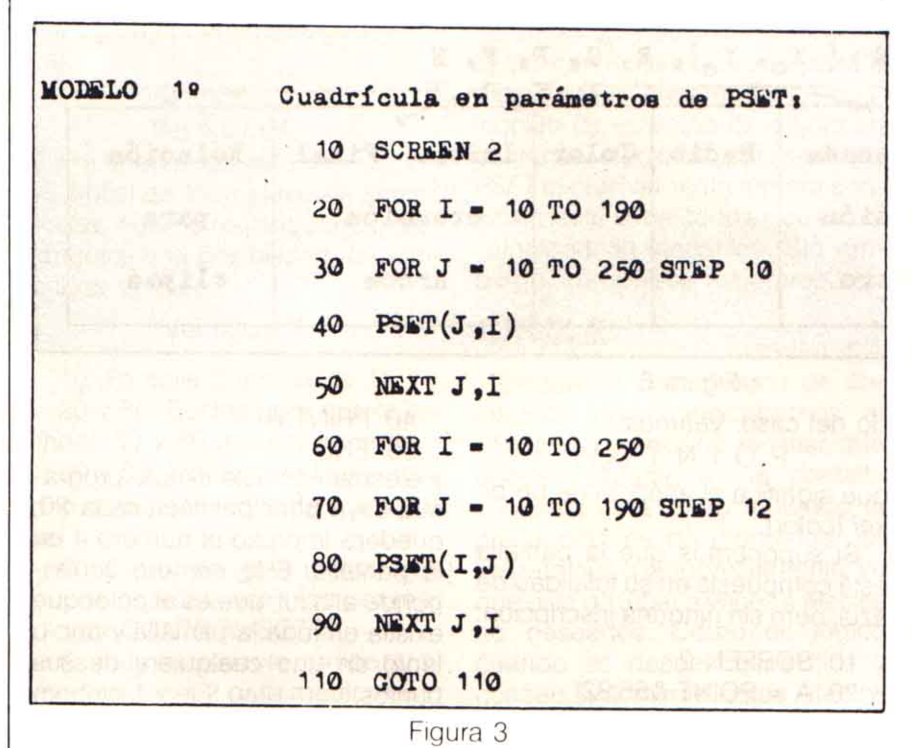
Se dispone también de un comando *PRESET* muy parecido al anterior con una diferencia sola-

SCREEN 2 : PSET(128,96) : FORT=0TO5000 : NEXTT

SCREEN 2	PSET(128,96)	FORT=0TO5000 : NEXTT
Pantalla de alta resolución(En SVI la 1)	coordenada x=128 y=96	Bucle de tiempo

Figura 2





MODELO 1º Cuadrícula en parámetros de PSET:

```
10 SCREEN 2
20 FOR I = 10 TO 190
30 FOR J = 10 TO 250 STEP 10
40 PSET(J,I)
50 NEXT J,I
60 FOR I = 10 TO 250
70 FOR J = 10 TO 190 STEP 12
80 PSET(I,J)
90 NEXT J,I
110 GOTO 110
```

Figura 3

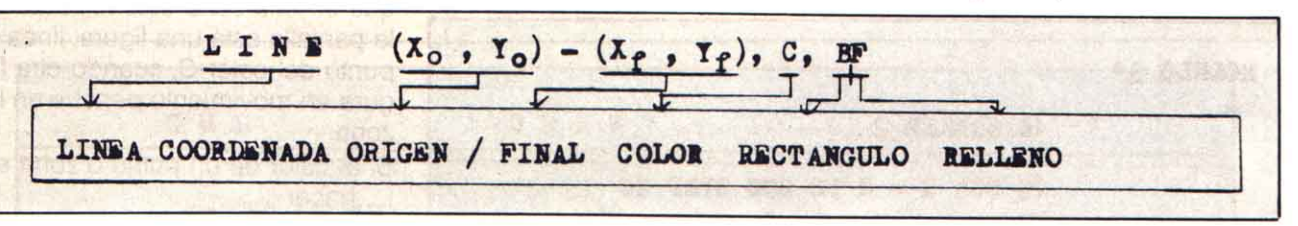
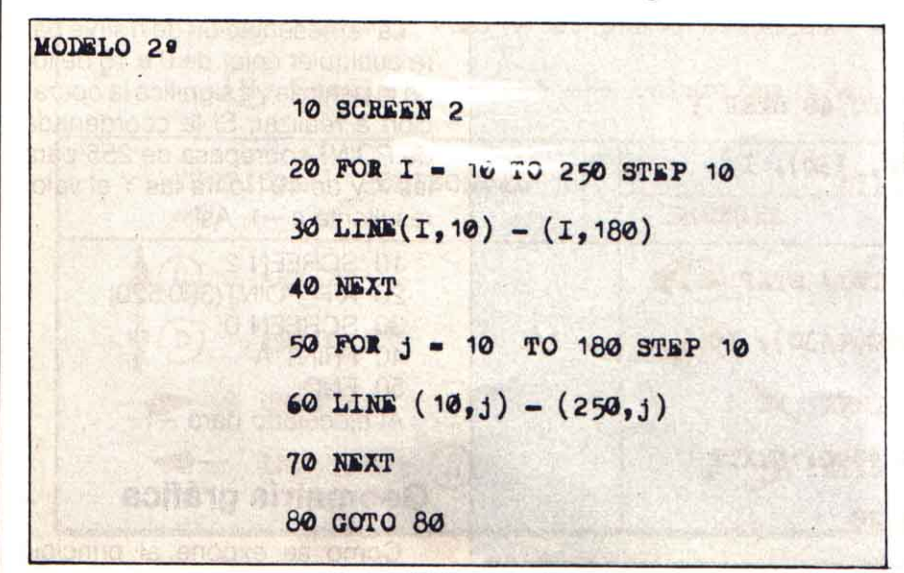


Figura 4



MODELO 2º

```
10 SCREEN 2
20 FOR I = 10 TO 250 STEP 10
30 LINE(I,10) - (I,180)
40 NEXT
50 FOR j = 10  TO 180 STEP 10
60 LINE (10,j) - (250,j)
70 NEXT
80 GOTO 80
```

Figura 5

mente relativa al tratamiento del color:

C = color de fonro para *PRESET*

y el algoritmo es idéntico al de *PSET:*

P R E S E T (X, Y), C

Por lógica si primero se escribe un *PSET* y a continuación un *PRESET* con las mismas X e Y, simulará un borrado, ya que sustituye el color de primer plano por el de fondo:

```
10 SCREEN 2
20 FOR I=80 TO 160
30 PSET(I,90)
40 NEXT
50 FOR J=80 TO 160
60 PRESET(J,90)
70 NEXT
80 GOTO 20
```

En primer lugar se inscriben puntos de izquierda a derecha más o menos en el centro de la pantalla y a continuación se «borran» en el mismo sentido. Líneas 30, para lo primero y 60 para lo se-

CIRCLE (X_c, Y_c), R, C, P, F, E

Cirounferencia Círculo Curva	Coordenada posición centro	Radio	Color	Inicio Final creación arcos	Relación para elipse

Figura 6

gundo. La lína 80 reenvía a la 20 para permitir la repetición del ciclo. Para verlo más lentamente introducir entre las líneas 40 y 50 un bucle de tiempo.

Condiciones

En muchas ocasiones da la impresión que un comando no tiene utilidad aparente. Sus posibilidades depende del análisis avanzado del caso. Veamos:

P O I N T (X,Y)

que significa el aspecto de un *pixel* (color).

Si suponemos que la pantalla está compuesta en su totalidad de azul, pero sin ninguna inscripción:

```
10 SCREEN 2
20 A = POINT &55,92)
30 SCREEN 0
40 PRINT A
50 END
```

y ejecutamos este simple programa, cuya principal línea es la 20, quedará impreso el número 4 en la pantalla. Este número corresponde al azul, que es el color que existía en toda la pantalla y por lo tanto en uno cualquiera de sus puntos.

¿Para qué sirve? Supongamos que en una zona determinada de la pantalla está una figura, línea o punto del color C, cuando otra figura en movimiento penetre en la zona:

Si el color de un punto o zona es ..... hacer ......

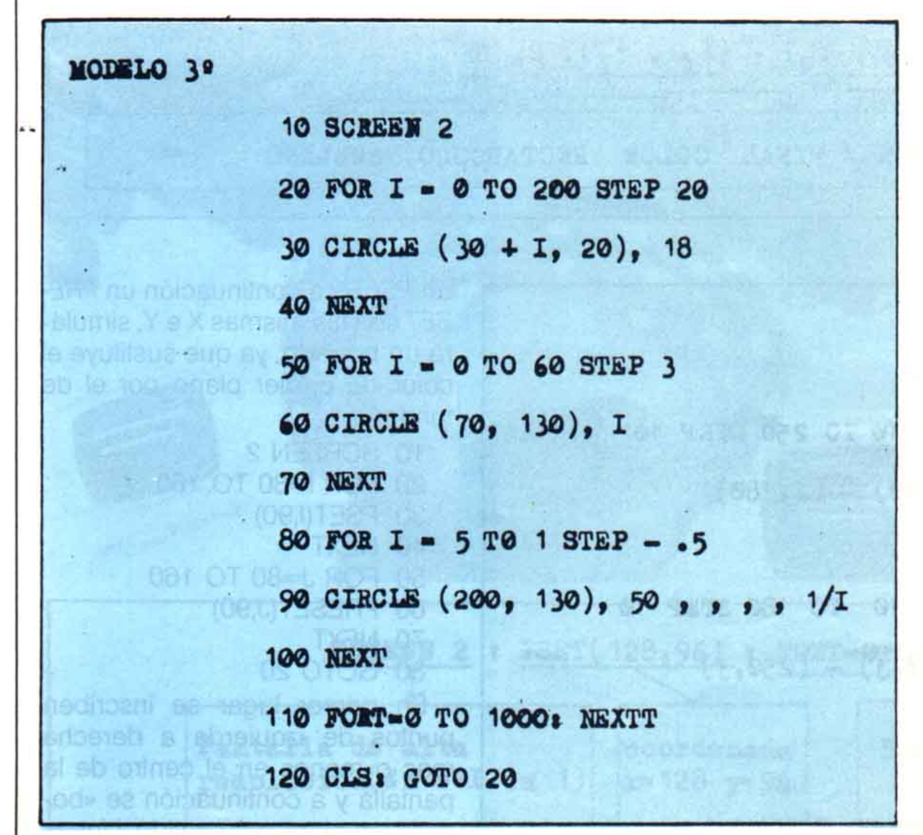
Es decir:

IF POINT (X,Y) = n THEN z

La representación de n sirve para cualquier color de 0 a 15 de toda la pantalla y z significa la operación a realizar. Si la coordenada de *POINT* sobrepasa de 255 para las X y de 191 para las Y el valor resultante s −1. Así:

```
10 SCREEN 2
20 A = POINT(300,520)
30 SCREEN 0
40 PRINT A
50 END
```

Al ejecutarlo dará −1.



```
MODELO 3º

10 SCREEN 2
20 FOR I = 0 TO 200 STEP 20
30 CIRCLE (30 + I, 20), 18
40 NEXT
50 FOR I = 0 TO 60 STEP 3
60 CIRCLE (70, 130), I
70 NEXT
80 FOR I = 5 TO 1 STEP -.5
90 CIRCLE (200, 130), 50 , , , , 1/I
100 NEXT
110 FORT=0 TO 1000: NEXTT
120 CLS: GOTO 20
```

Figura 7

Geometría gráfica

Como se expone al principio vamos a comentar los métodos

principales para figuras geométricas.

En primer lugar:

Ver figura 4

Por medio de esta instrucción se obtienen toda clase de líneas, rectas, figuras rectangulares, rectángulos y la posibilidad de rellenarlos.

Ver figura 5

10 Pantalla 2 (en SVI la 1).

20 y 50 Bucles para repetir las líneas 30 y 60 respectivamente.

30 y 60 Trazado de líneas que dan como resultado una cuadrícula.

80 Cierre en goto infinito que puede abrirse a voluntad por medio de *CONTROL/SOTP.*

Es muy interesante comparar el modelo 1 y el 2 para medir más o menos la velocidad de ejecución.

En lugar de 10 en los *STEP* de las líneas 20 y 50 pruébese con 5, 2, etc. con lo cual se aumentará o disminuirá el entramado. La confección de este tipo de dibujo en contra del efectuado por medio de *PSET* exclusivamente mejora sensiblemente sobre todo en cuanto a velocidad de ejecución. Sin embargo, aún puede mejorarse.

CIRCLE

Seguimos con el resto de comandos como indicábamos al principio y conviene resaltar que pueden situarse en la pantalla conjuntamente, pero teniendo la precaución de no montarlos por sus vértices o tangencialmente, ya que esto da lugar a veces a efectos no deseados. Como es lógico cuando se deseen específica y consecuentemente estos efectos el método de prueba es el único que aconsejamos.



La fórmula de las figuras curvas en general se refleja por medio de:

20 CIRCLE(127,90), 65,,,, 1.333

por ejemplo, pero es imprescindible anteponer una *SCREEN* de dibujo:

10 SCREEN 2

y para dejar fijo el dibujo hasta la

G M L — **D R A W " .... "**

(B) Mueve (Sin trazo) (M) Mueve X, Y (0→255, 0→191)

(N) Mueve y vuelve al punto original

(S) Escala (0→255 --- relación: s/4) (C) color (0→15)

VERTICAL HORIZONTAL		DIAGONAL DERECHA		DIAGONAL IZQUIERDA
↑ (U)	ARRIBA	↗ (E)	ARRIBA	
↓ (D)	ABAJO	↘ (F)	ABAJO	
→	DERECHA (R)		ARRIBA ↖	(H)
←	IZQUIERDA (L)		ABAJO ↙	(G)

Figura 8

actuación de *CONTROL/STOP:*

30 GOTO 30

El coeficiente 1.333 debe variarse según la forma deseada y esto es indicativo en cada caso. Por ello la fórmula general se expone así:

Ver figura 6

Para corregir la diferencia entre TV/monitor y coordenadas se puede variar adecuadamente el punto E. En el ejemplo clave anterior —1.333—.

Ver figura 7

10 Pantalla alta definición (en SVI 318/328 la 1).

20, 50 y 80 Bucles diversos.

30 Círculos iguales en distintas posiciones.

60 Círculos concéntricos en la misma posición central.

90 Elipses concéntricas o casi concéntricas.

```
MODELO 4º

 10 COLOR 5, 15, 15: DRAW"AO": SCREEN 2
 20 FOR E = 10 TO 255 STEP 5
 30 DRAW "BM 20,171; S = E; C2URDL"
 40 NEXT
 50 FOR E = 255 TO 10 STEP -5
 60 DRAW " BM 245, 10; S = E; C8DLUR "
 70 NEXT
 80 FOR E = 10 TO 255 STEP 5
 90 DRAW " BM 20, 10; S = E; C4DRUL"
100 NEXT
110 FOR E = 255 TO 10 STEP -5
120 DRAW " BM 245,171; S =E; C10ULDR"
130 NEXT
140 FOR I = 3 TO 0 STEP -1
150 DRAW " BM 125,86; C1R5D10L5U10 A = I;"
160 FORT = 0 TO 500: NEXTT
170 NEXT
180 GOTO 180
```

Figura 9

Para salir de la repetición sin fin:

CONTROL/STOP

DRAW (dibujo libre)

El lenguaje gráfico GML tiene como pantallas exclusivas las 2 y 3, por ello hay que definir cualquiera de las dos previamente al empleo de la macroinstrucción *DRAW*, que por cierto es en realidad una variable alfanumérica especial. La palabra reservada será por lo tanto:

DRAW "................... // ...."

Draw indica dibujo o trazo y las comillas encierran una cadena o *string*.

También se puede emplear:

z$ = ".........." w$ = ".........."

y luego:

DRAW z$; w$

Este procedimiento permite una gran flexibilidad en dibujos parciales ya que se pueden tomar, por ejemplo, la z$ tres veces y la w$ una o viceversa. Así como tomar partes de un dibujo que luego se pueden recomponer libremente.

Ya vimos la introducción de coordenadas por medio de *PSET*, pero en GML es mucho más interesante.

D R A W "BM 127,80 ...."

indicando el origen del dibujo posterior como ahora veremos:

Ver figura 8

Cada letra se acompaña del número total de veces que se va a trazar la línea. Si fuera sólo una vez no es necesario escribirlo.

DRAW "BM 100,50" Situación x= 100 :: y=50 No hay trazo.
DRAW " S100" Escala 25 (ya que 100/4 = 25).

DRAW "U5R10" Angulo formado por 5 pixels hacia arriba y. 10 hacia la derecha.
DRAW "C12" Color verde (12).

Con estos ejemplos podemos hacernos una idea de la construcción de este especial lenguaje (GML). Ya veremos programas modelo después.

La capacidad de este proceso podemos resumierla:

1) Realiza cualquier dibujo.
2) Puede saltar a otro punto distante sin trazo.
3) Aumento o disminución del tamaño del dibujo (efectos zoom).
4) Vuelta al punto inicial para trazar otra vez (abanico).
5) Colores de espacio limitado.

Otros procedimientos pueden superar al GML, desde luego fuera del propio sistema, pero este enseña a manejar cadenas, pantallas y crear nuestros propios programas de dibujo.

Ver figura 9

En las línas 30, 60 y 90 se comprende el empleo directo de



DRAW en cuanto a las coordenadas de inicio de cada fase (BM).

S = E; establece una referencia con respecto al bucle que corresponda según la escala que llega hasta 255 como se ve en cada FOR.

C representa el color deseado. Mírese la tabla de DRAW.

En la línea 150 se apunta:

A = I;

(ES INDISPENSABLE EL PUNTO Y COMA ;) que corresponde exactamente a cada una de las 4 posiciones angulares 0 a 3 de 90° de cualquier dibujo. Hemos incluido un bucle de tiempo en la línea 160 para observar mejor este efecto. Es muy conveniente que prestemos atención a este detalle para emplearlo convenientemente.

PAINT

Es totalmente imprescindible que PAINT:

–SOLO SE PUEDE UTILIZAR EN DIBUJOS CERRADOS–

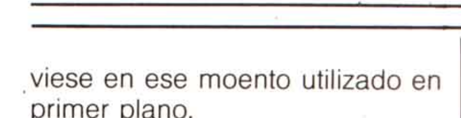
Lleva la siguiente asignación:

PAINT (X,Y), C

en el que X e Y son la coordenada del inicio dentro del sector que se prenteda llenar y C el color que debe ser el mismo que las líneas o bordes que lo limiten. El color del borde sólo es utilizable en *SCREEN 3* añadiéndolo a la citada asignación después de otra coma (,).

```
10 SCREEN 2
20 CIRCLE (127,80), 50
30 PAINT (127,80)
40 GOTO 40
```

Al no indicarse color alguno, se coloreará del mimo en que estu-



viese en ese moento utilizado en primer plano.

En el caso de *CIRCLE* puede sustituirse la línea 30 por:

30 PAINT STEP (0,0)

ya que desde el punto central siempre se está dentro del círculo. *STEP* instruye al programa para saltar desde la posición anterior X e Y hasta la indicada. Al ser (0,0) en este dado se que da en la precedente, con lo cual se gana cierto tiempo.

La lentitud de *PAINT* es a veces no permutable, pero hay que agotar los pasos para llegar a su utilización directa.

Por ejemplo:

```
10 SCREEN 2
20 LINE(50,100)–(200,50),,B
30 PAINT (100,75)
40 GOTO 40
```

la ejecución produce un rectángulo que se rellena por medio de la línea 30, de tal modo que es 10 VECES más lenta que:

```
10 SCREEN 2
20 LINE (50,100)–(200,50),,BF
30 GOTO 30
```

ya que el relleno está preestablecido en F tal como indicamos al referirnos a la instrucción *LINE*.

José Leal Rodríguez

SOFTWARE

Programa: The Dam Busters
Tipo: Juego
Distribuidor: ERBE
Formato: Cassette

Pocos son los juegos de simulación existentes en el mercado. De ellos, *The Dam Busters* es el más realista, complicado y uno de los mejores que hemos probado.

Basado en hechos de la II Guerra Mundial, *Dam Busters* es un simulador de vuelo en el que tu misión consiste en destruir una serie de presas volando a baja altura, de noche y con luna llena. La perspectiva no parece muy buena, ya que las baterías antiaéreas permanecen siempre alerta ante cualquier situación anormal, por este motivo deberás tener un control absoluto sobre tus hombres y equipo.



Para empezar, tienes dos opciones a elegir; 1) Teniente de vuelo ó 2) Jefe de Escuadrón. Una vez elegida la que más guste, tendrás que familiarizarte con los tripulantes y la misión dentro del contexto global de la operación. Pulsando la teclas 1 a 7 podrá controlar los siguientes elementos:

1. Piloto.
2. Artillero delantero.
3. Artillero trasero.
4. Bombardero.
5. Navegador.
6. Pantalla del Primer Ingeniero.
7. Pantalla del Segundo Ingeniero.

Cada uno de ellos tiene una labor concreta y una pantalla específica. Es decir, en el momento en que se pulse una opción aparecerá la pantalla de la que se ha escogido con todos sus elementos.

En la primera de ellas, el piloto controla la dirección del avión con el *joystick*, de la misma manera que esta palanca funciona en un avión. Empujando la palanca, el avión baja. Tirando de ella, el avión sube y para girar hacia la izquierda o derecha, haremos lo propio con el *joystick*. Con la opción siguiente, se controla la función del artillero delantero, que con dos ametralladoras dobles será el encargado de ir despejando el camino. Como es lógico, las ametralladoras se dirigen con el *joystick*. Además de controlar las ametralladoras, también hace las funciones de bombardero, aunque tiene poco sentido acceder a esta opción si no es cerca del objetivo final. La siguiente opción, es la del artillero trasero. A la vez que el artillero delantero, este controla dos ametralladoras dobles y se maneja con el *joystick*. Luego está el navegador, que tiene la misión más importante, la responsabilidad de buscar la ruta a través del territorio enemigo hasta las presas. En su pantalla, aparecerá un mapa en el cual hay dos objetivos que se mueven. Uno muestra la posición actual del avión y otro, el cursor de navegación que se usa para elegir la dirección en compás. Por último quedan las pantallas del ingeniero, que es el encargado de vigilar los motores, tanto su desarrollo, como evitar los posibles incendios y pasadas de vueltas que suelen ocurrir en estos casos.

El programa viene acompañado por un manual en el que se explican, tanto las características del avión que se emplea en la misión, como los puestos y misión de cada tripulante, así como la bomba utilizada para este tipo de operaciones, pues al ser especial requiere unos conocimientos previos antes de lanzarla.

Un buen programa que, necesita horas de entrenamiento para poder dominar el arte de volar.

Puntuación:
Presentación: 8
Claridad: 8
Rapidez: 9
Adicción: 8

Programa: BATMAN
Tipo: Juego
Distribuidor: ERBE
Formato: Cassette.

Te encuentras en las Bati-Cuevas, debajo de Gotham-City. Robin ha sido capturado por los enemigos de Batman. La única posibilidad de escapar es emsamblar el Baticraf, cuyas piezas se hallan dispersas por las catacumbas.

Si tuviéramos que hacer una evaluación de lo que ha supuesto estos últimos meses en el mercado de los juegos de arcade-aventura, nos parece verdaderamente imposible que se pudiera llegar a mejorar significativamente un juego de estas características, dadas las limitaciones que supone a estas alturas disponer de tan sólo 64K de memoria para la realización de este tipo de obras de arte en el mundo del software.

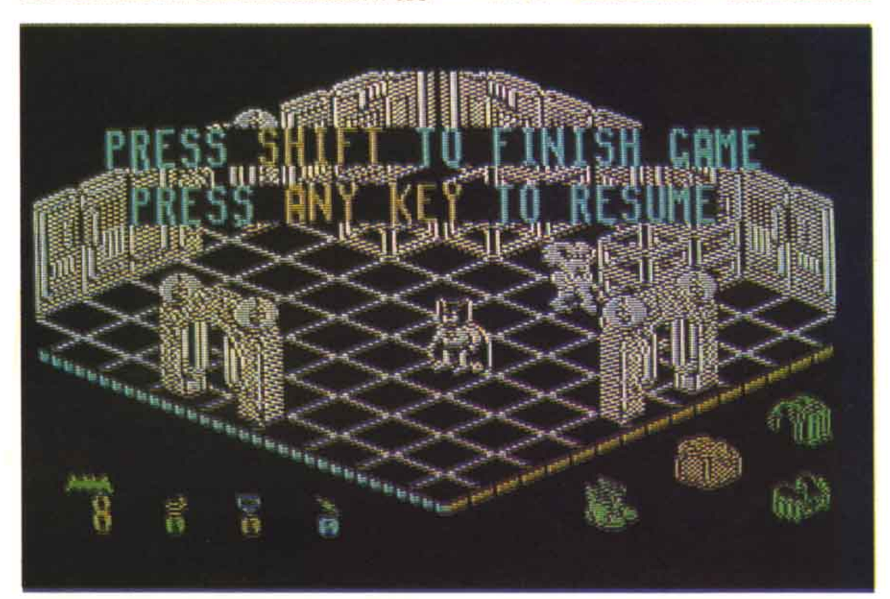
Para aquellos conocedores de la técnica de Ultimate, podemos decir que este juego sigue una técnica muy similar a los anteriores pero en el momento en que nos adentremos en él, comprobaremos que no sólo existen técnicas, gráficos, versatilidad de movimientos, etc., sino incluso en la trama que sigue, ya que es muy interesante.

Podemos referirnos a Ocean como un genio que ha conseguido lo irrealizable por el momento.

Comenzamos por tanto la aventura, señalando los primeros pasos que debemos seguir. Hay que partir en un primer momento en busca del equipo de Batman. Son cuatro objetos imprescindibles que deben ser encontrados lo antes posible para poder desarrollar sus poderes; las batibotas, sirven para saltar en los momentos en que nos veamos más asediados. La Batimochila, imprescindible, ya que a lo largo de la aventura tendremos que recoger numerosos objetos importantes para nuestra supervivencia.

Batimotor, indispensable para controlar nuestro vuelo, y la velocidad de éste, así como el Baticinturón, que reducirá nuestra velocidad en las caidas. Sólo sobrevivir en estas condiciones hay algunos Batipoderes repartidos por las catacumbas en formas de pequeños Batmans, estos pueden darnos vidas extras, energía, inmunizarlos a los peligros o poder saltar más alto, pero no todo es de color de rosa, existen unos falsos Batipoderes, que en caso de confundirnos neutralizarán todos aquellos que hayamos conseguido antes.

Otro elemento interesante



y únicamente habiendo reunido estos objetos, se podrá proseguir con alguna probabilidad de éxito, ya que la misión no consiste en esto tan solo, sino en encontrar cada una de las piezas que forman el Baticraf para poder ir en la ayuda de Robin y rescatarlo.

¿Cuáles son los peligros que nos acechan?, son numerosos y por tanto debemos estar a cada momento con todos nuestros sentidos en movimiento.

Esta odisea nos preparará trampas en las que nos será muy difícil no caer, pero no hay nada imposible y enemigos con los que mejor no nos hubiéramos encontrado. Para ayudarnos a son las Batiseñales: tocándolas nos permitirán comenzar desde ese punto con el mismo número de vidas y objetos que transportemos, si al perder la partida no habiendo alcanzado nuestro fin, deseamos continuar sin tener que volver a empezar de nuevo.

El juego es muy completo en cuanto a su presentación, opciones de menú, etc., el nivel técnico es muy elevado, y han sabido aprovechar en gran manera la perfección de la capacidad gráfica del ordenador.

Presentación: 9
Claridad: 7
Rapidez: 9
Adicción: 10

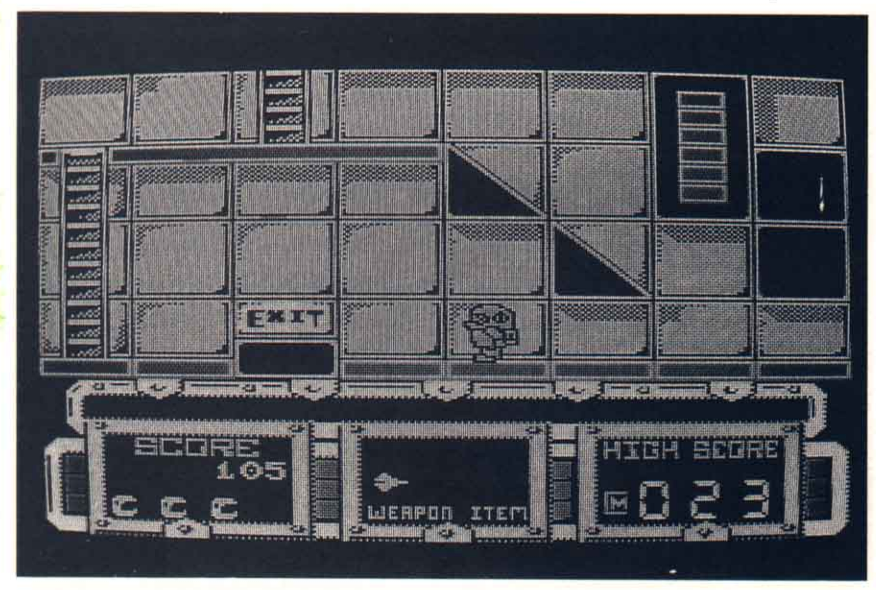
SOFTWARE

Programa: Future Knight
Tipo: Juego
Distribuidor: ERBE
Formato: Cassette

Numerosos son los diferentes juegos que han tomado como basé el entrañable juego de los comecosos, como habitualmente lo conocemos, pero todos ellos, no han sabido crear partiendo de una base inmejorable como es la más sencilla estructura de un juego de software.

Future Knight, es una sofisticación de este tipo de juegos aunque no podemos denominarlo como tal, ya que este juego en sí es una combinación de varios, extrayendo de cada uno lo mejor.

No podemos hablar de pantallas, ya que es una sucesión de movimientos que debemos de efectuar a lo largo de un entramado de escaleras, niveles y módulos los cuales debemos sortear y examinar, pues la misión comenzará en el momento en que hayamos conseguido todos los objetos necesarios para enfrentarnos a los variados y múltiples enemigos que se nos presentan. Equipados con una sofisticada indumentaria, un traje espacial, seremos inmunes a muchos enemigos desde el momento en que comience la partida, ahora bien, si conseguimos batirlos con nuestra pistola de torpedos, además de conservar nuestas fuerzas, conseguiremos puntos. El traje en sí nos permitirá movernos por cualquier sitio siempre ascendiendo o descendiendo en busca de los elementos que lo forman, pistola de rayos laser, protector, etc., pero tan solo tendremos 900 sg para conseguir realizar esta primera prueba ya que cada vez ascendamos a otro nivel los enemigos serán más audaces y veloces.



Como controlar a nuestro astronauta, si utilizamos joystick, no existe problema alguno, si por el contrario nos gusta más el teclado existen teclas designadas, para realizar funciones en caso de ayuda que os serán muy útiles. En primer lugar debéis echar un vistazo para familiarizaros con los peligros existentes. El juego, a diferencia de otros no se encuentra sobrecargado de elementos que os hagan la vida imposible, es por ello que alcanzar vuestro objetivo no será difícil. Calaveras, moles, máquinas y verdaderos robots destructores saldrán a vuestro encuentro, disparad y luego proseguid, de esta forma no agotaréis vuestra energía y sumaréis puntos.

Inspeccionad el piso inferior, e id ascendiendo a medida que vais encontrando enemigos, sobre todo daros prisa pues el tiempo se acaba y recoger todos los objetos con que os encontraréis.

La pantalla, en su parte inferior consta de un reloj digital que os irá recordando el tiempo que os queda, los elementos que habéis conseguido y las vidas que os quedan.

Con estas pequeñas connotaciones podéis haceros una idea de las características del juego, pero creemos que estas palabras son pocas para referirnos a este juego. Excelente en su creación de gráficos, los detalles están muy mejorados y el sonido no interrumpe en absoluto en su desarrollo, asimismo todos los elementos que forman el juego son especiales, es decir, no son los elementos clásicos que estamos acostumbrados a ver en estos juegos, tan sólo, claro está, a nuestro protagonista, y a modo de sugerencia esto también se podía modificar.

El nivel de dificultad no es elevado pero la rapidez y exactitud sí. ¡¡Adelante!!

Puntuación:
Presentación: 8
Claridad: 6
Rapidez: 7
Adicción: 8

Programa: Avenger
Tipo: Juego
Distribuidor: ERBE
Formato: Cassette

Avenger, para aquellos adictos de los juegos de rapidez y misterio, es sin duda uno de los mejores prototipos actualmente.

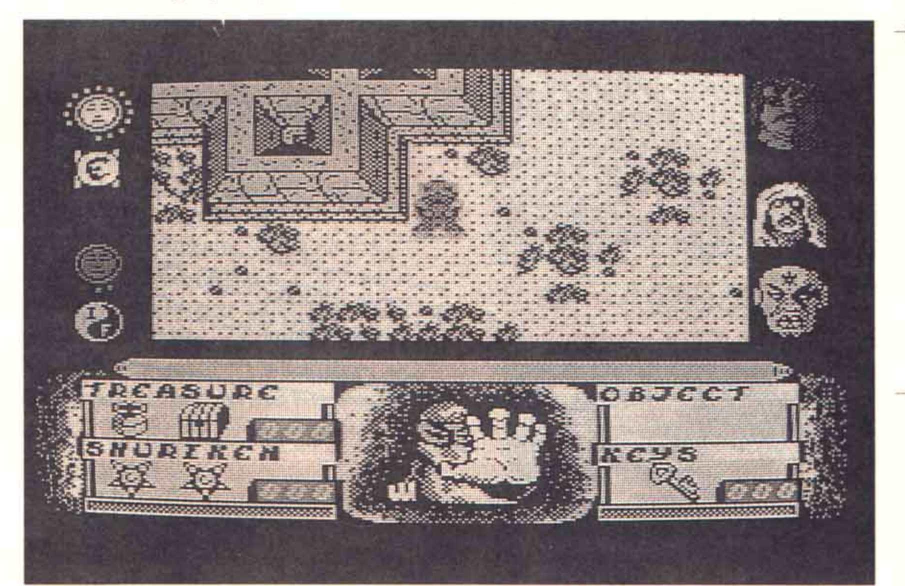
Las bases de creación que han utilizado es como en todos los que hasta hoy en día hemos conocido, sencillas, pero esto no quiere decir simple y esta es la principal característica de este jujego. El molde, que pudiéramos decir han utilizado para crearlo, es el de multitud de juegos de arcade-aventura, que nos inundan el mercado, pero esto nos permite la posibilidad de ver antes de creer.

El juego en sí tiene ciertas individualidades que le hacen característico y a su vez innovista en cuanto al tipo de gráficos, los cuales debemos destacar.

El mapa del juego no es extenso, ya que nos presenta a modo de croquis todo el escenario en que se desarrolla. Los laberintos y pasadizos de un recinto medieval configuran el escenario de la aventura. La principal diferencia que encontramos referido a esto, es que las pantallas no son sucesivas cámaras de las que pasas, o no sales, sino la viabilidad de un recinto el cual tiene sus normas de entrada pero no de salida, tan sólo tu habilidad y velocidad. Objetivos irán saliendo a tu paso, pero debes tener en cuenta el tiempo. Tendrás tan solo 10 soles, forma que se usaba en la antigua Edad Media para medirlo. Encontrar el tesoro es el principal objetivo de todos los aventureros, pues aquí hay dos, pero también existen otros elementos que deben ser considerados como tal, las llaves, sin ellas no podrás acceder a ningún sitio, ya que cada cámara a la que entres necesitarás abrir su puerta correspondiente, aunque esto no es para todas igual. Significa que consiguiendo abrir la primera puerta todas las que forman ese recinto serán derribadas con un acertado golpe, de Karate ya que nuestro amigo en sus numerosos viajes de aventuras ha asimilado las más eficientes técnicas.

No tendrás problemas en cuanto a los peligros que se te avecinan pues tú eres demasiado rápido para los moradores del castillo, arañas, monstruos, y verdugos, los cuales no te matarán sino que harán agotar tu tiempo hasta que pierdas y desaparezcas. La mejor forma de batirlos es, o bien lanzando flechas de fuego ya que no es difícil acertarles o enfrentándote con ellos con tus espectaculares golpes de karate, y sal corriendo ya que cada vez que mates a uno no conseguirás puntos pues el juego consiste en recorrer el mayor tramo de aventura.

Las propias características del juego, que principalmente son el entretenimiento y con ello la variedad y singularidad, sin duda se encuentra en sus gráficos. No podemos decir que sea una creación de las más avanzadas técnicas, pero sí de las más cuidades dentro de sus posibilidades. Un ejemplo a destacar dentro de los elementos gráficos que aparecen son unas pequeñas llamas de fuego que se encuentran en el suelo de los recintos y que nos impiden el paso, a esto le podemos unir la imaginación a la hora de crear los adversarios, su originalidad y sobre todo el color, fuerte pero que resalta aún más el juego. Unos consejos para jugar mejor, procurad ir bordenado la muralla ya que los caminos por fuera del recinto son muy anchos. Mucha paciencia si jugáis con joystick.



Puntuación:
Presentación: 8
Claridad: 8
Rapidez: 9
Adicción: 8

el mejor softwa





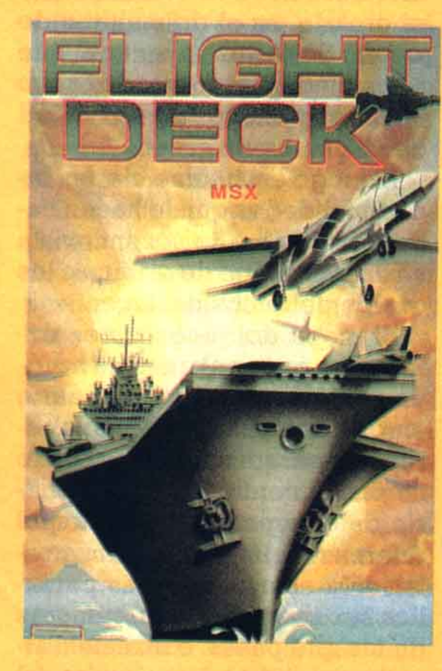
DROME

Entretanto en DROME, un Super-ordenador, debes encontrar y eliminar los sofisticados sistemas de defensa y supervivencia.

Has de elegir uno de los cuatro sectores que constituyen los mecanismos de defensa de esta terrorífica máquina.

Un atractivo juego de acción, donde se pone a prueba la capacidad de la máquina y del jugador.

Precio de venta 2.000 ptas. (IVA incluido)



FLIGHT DECK

Sienta la emoción del golfo de Sidra en casa.

FLIGHT DECK es un juego de estrategia y habilidiad en el que tendrás que desmantelar las bases enemigas.

Al mando de un portaaviones donde dispones de 10 unidades de combate... y poco tiempo.

Precio de venta 2.000 ptas. (IVA incluido)





MC-ATTACK

Ayuda a Fredy, el Rey de la Hamburguesa a preparar el suculento manjar que hace las delicias de los comensales.

Ten cuidado con las salchichas grasientas y los huevos escurridizos que intentarán arruinar tu exquisito plato.

Defínete con la pimienta y procura hacer el mejor número de hamburguesas posible.

... Buen provecho.

Precio de venta 750 ptas. (IVA incluido)

re para MSX



CONFUSED?

Es el puzzle electrónico.

El objeto del juego es resolver 10 puzzles con distinto número de piezas, a elegir, pero todas... MOVIENDOSE.

Pon a prueba tu inteligencia y capacidad de deducción para solucionar algunos de estos entretenidos rompecabezas.

Precio de venta 2.000 ptas. (IVA incluido)

NORTH SEA HELICOPTER

Una explosión en una plataforma en el mar del Norte arroja a los hombres a un destino incierto...

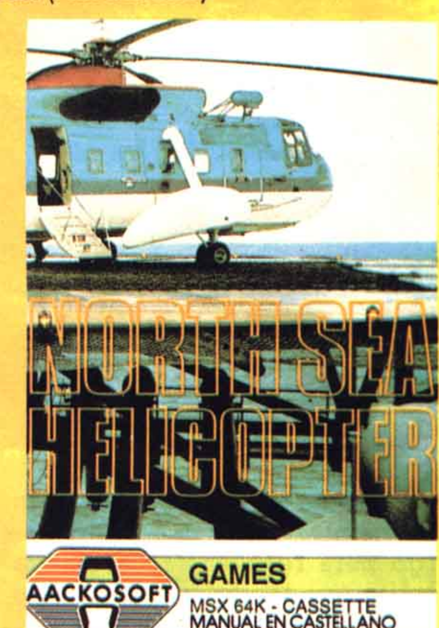
Empieza una carrera contrareloj para salvarles de su fatal situación.

Tienes que convertirte en un piloto experimentado para mantener el control del helicóptero... El tiempo empeora.

¿Crees que cumplirás la misión?

Precio de venta 2.000 ptas. (IVA incluido)







SPACE RESCUE

Desciende de tu nave nodriza a la superficie lunar e intenta rescatar a los hombres extraviados.

Ojo con los meteoritos que te destruirán cuando intentes regresar.

Disponer de nuevas plataformas para culminar con éxito la misión.

Desgraciadamente tu nave nodriza está bloqueada por unidades de combate enemigas... Intenta destruirlas.

Precio de venta 750 ptas. (IVA incluido)

infodis, s.a.

Solicita tu programa favorito en tu tienda especializada habitual o recorta y envía este cupón a INFODIS, S. A., c/Bravo Murillo, 377. 5.° A - 28020 MADRID.

NOMBRE ____________

DIRECCION ____________

POBLACION ________ PROVINCIA ________ C. P. ________

Forma de pago: CHEQUE ☐ CONTRAREEMBOLSO ☐

TARJETA DE CREDITO: American Express ☐ VISA ☐ INTERBANK ☐

Número de mi tarjeta ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

CADUCIDAD ____________

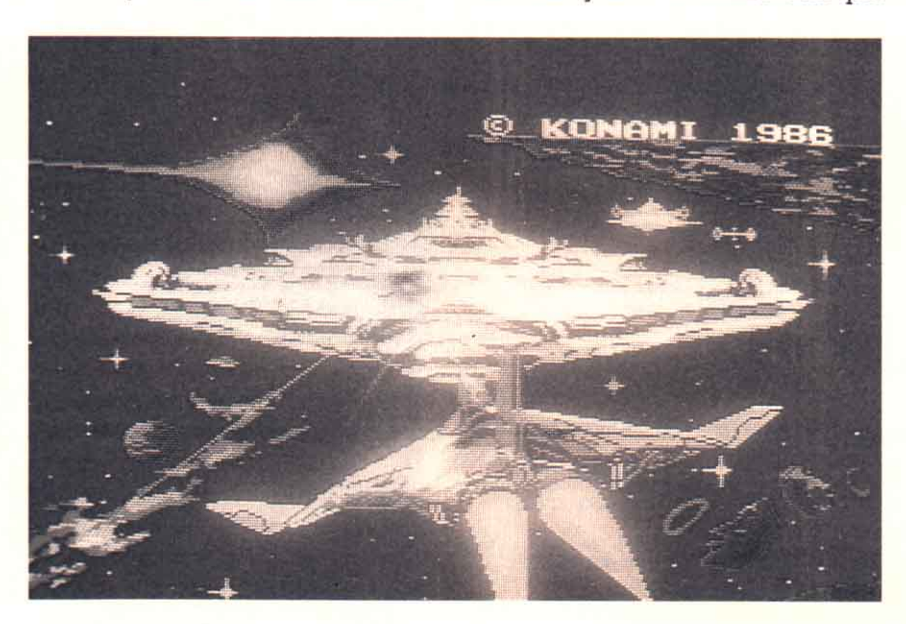
Firma ____________

TITULO	CANTIDAD	TITULO	CANTIDAD
DROME FLIGH DECK MC-ATTACK		CONFUSED? NORTH SEA HELICOPTER SPACE RESCUE	

SOFTWARE

Programa: Nemesis
Tipo: Juego
Distribuidor: SERMA
Formato: Cassette

Estamos ante un programa que promete ser el juego del año. La diversidad de las pantallas, el nivel de los gráficos, el sonido y todas las características de los programas *«made in Konami»*, se juntan en este juego para realizar una auténtica obra que tendrá a más de uno pegado a la silla una hora sí y otra también.



Las pantallas iniciales, sirven para crear ambiente y para hacerse una idea de lo que nos espera. Hay que disparar a todo lo que salga por tierra y aire, y dejar los menos enemigos posibles puesto que, como buenos marcianos, dispararán tanto hacia adelante como hacia atrás, hacia arriba y hacia abajo, lo que significa que habrá que tener los ojos puestos en todos los lados y procurar esquivar los tiros, las bombas, etc.

Se empieza con tres naves y con la opción de uno o dos jugadores. Una vez realizada la elección correspondiente, hay que empezar a disparar con orden para conseguir una serie de elementos que nos ayudarán en el cometido de nuestra misión. Es muy útil dejar que el programa nos muestre la demostración y así hacerse una idea de lo que va a ser y de las armas con que se cuentan, ya que aunque inicialmente sólo podamos disparar tiro a tiro, cuando hayamos recorrido algunas pantallas podos tener hasta tres naves (la propia y dos sombras) disparando ráfagas de rayos laser y lanzando bombas a la vez, que destruyen todo lo que pillan por el camino. Estas opciones se encuentran indicadas en la línea inferior de la pantalla y son: *speed up, missile, doble, laser, option y ?.* Cada una de ellas tiene un cometido especial que veremos a continuación. La primera de ellas, *speed up,* permite conseguir más velocidad de la que inicialmente cuentas, muy útil a la hora de esquivar los pesados marcianos que nos persiguen. Con *missile,* podrá lanzar bombas sobre los enemigos que se hallen en tierra. Esta opción facilita la concentración en el vuelo, al poder desentenderse de los que se encuentran bajo nosotros. La tercera opción es *doble,* con la que se pasará del tiro simple al tiro en dos direcciones distintas, con esta se eliminarán los enemigos que vengan por arriba. La siguiente es el *laser,* muy importante por el barrido que realiza con cada tiro. Con *option,* podremos contar con una nave sombra, que nos seguirá a todas partes y disparará a la vez que nosotros. Es indestructible y muy útil, pero hay que saber utilizarla. Por último, queda la *?,* que es un escudo protector, útil por razones obvias.

Estos son los elementos necesarios para completar la misión, que está compuesta por varias etapas, al final de cada una de ellas hay que enfrentarse a la super-nave enemiga cuyos 6 cañones nos estarán esperando. Para eliminarla, habrá que disparar repetidas veces al centro y no desanimarse. Y no os explicaremos más, porque si no sabríais más que nosotros. ¡¡Suerte y feliz caza!!

Puntuación:
Presentación: 9
Claridad: 9
Rapidez: 10
Adicción: 10



GANE
7.000 PTAS.
todos los meses
PARTICIPANDO EN NUESTRO CONCURSO
MSX Magazine premiará cada mes los programas que nos hagan llegar nuestros lectores.
Para participar en este concurso abierto, todo aficionado a los ordenadores con este estándar deberá hacer llegar a la redacción de la revista el listado, un cassette y un texto explicativo.
Entre todos los programas que recibamos cada mes, serán seleccionados para su publicación aquellos que reúnan los siguientes criterios:
— Originalidad de la aplicación.
— Simplicidad del método de programación.
La única condición para participar en el concurso será que los programas no hayan sido publicados previamente en ninguna revista.
Enviar vuestros programas a: MSX Magazine
C/Bravo Murillo, 377 - 5.º A 28020 MADRID

SOFTWARE

Programa: Idea-text
Tipo: Aplicación
Distribuidor: Idealogic
Formato: Cartucho ROM

A pesar del tiempo transcurrido, Idea-text, no deja de ser novedad. Cuando los ordenadores MSX de la I Generación pedían a gritos programas de aplicación y/o gestión, aparecieron los primeros trabajos de Idealogic en cassette. Ha pasado mucho tiempo desde entonces, y este programa no ha dejado más que un buen sabor de boca. Es sencillo, fácilmente manejable y muy completo. Prueba de ello es que lo hemos empleado en varios trabajos, saliendo airoso de los compromisos.

Al conectar el cartucho, este detecta si la unidad de discos está conectada, con lo cual se altera la línea inferior del menú de la pantalla principal y donde antes aparecían las funciones:

BLOQ. CINTA BUSQ. CONT. VSLR.

ahora aparece:

BLOQ. DISCO BUSQ. CONT. VSLR.

con lo que este aspecto queda totalmente de la mano del ordenador.

Dentro de la pantalla principal, hay una ventana de información en la parte inferior de la pantalla que es la que indica en todo momento el estado del trabajo, devolviendo mensajes y errores si fuera necesario. Normalmente, tendrá la siguiente información:

LIN: indica el número de línea de pantalla en que se encuentre el cursor.

COL: indica la columna de pantalla en que se encuentra el cursor.

MEM: es la memoria que queda disponible para texto (42500 caracteres como máximo en un MSX con 64K).

Esta información, junto con la proporcionada por las teclas de función F1 a F5, permiten tener un fácil acceso a todo tipo de datos mientras se está escribiendo.

Después de esta breve introducción al programa, es importante resaltar que en la pantalla se verán 29 caracteres por línea, sin embargo, la opción F5 (VSLR) le permitirá observar en pantalla como quedaría el listado. Es decir, si al definir el formato de salida del documento, hemos indicado que se imprima a 80 columnas, estas sólo se verán al pulsar la opción F5, en la cual, el usuario comprobará como queda el trabajo final.

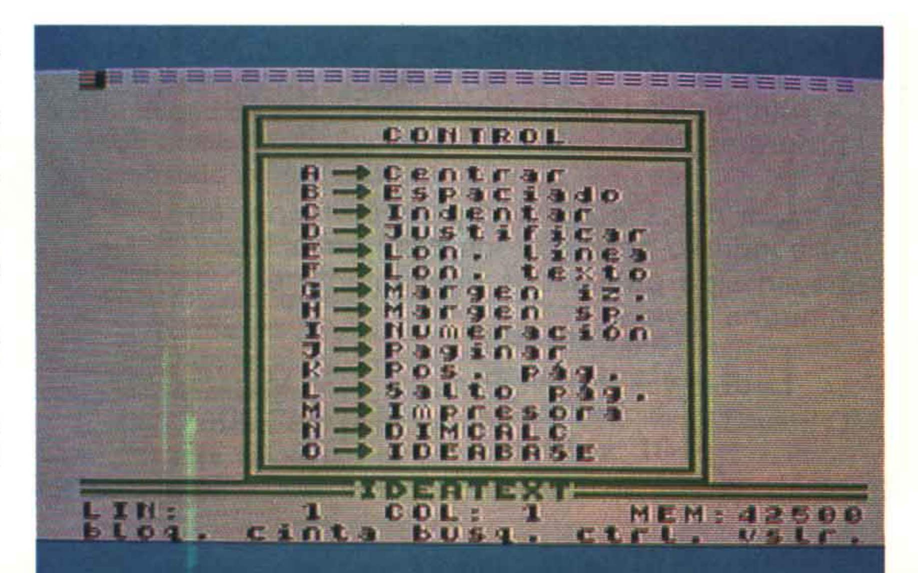
Veamos lo que realiza cada tecla de función. La F1 (BLOQUES), gestiona bloques de texto. Estos se marcan, al principio y al final, y se tratan como una unidad, pudiéndose borrar, mover, copiar, grabar, insertar y limpiar.

La F2 (DISCO o CINTA), permite realizar cualquier operación con uno de estos dos periféricos. En el caso de utilizar el disco, tendríamos la posibilidad de grabar, recuperar, borrar, renombrar y mostrar el directorio del disco. Sin embargo, si fuera la cinta el soporte utilizado, tendríamos las siguientes opciones: grabar, recuperar y verificar. La tecla F3 (BUSQ.), permite la búsqueda de textos y etiquetas. Sus opciones son: poner etiquetas, salta etiquetas, busca una cadena de texto y sustituye una cadena de texto.

La tecla F4 (CONTROL), sitúa en el texto los caracteres de control que permiten formatear la salida impresa. En esta opción, podemos introducir todo tipo de elementos que ayuden a obtener una impresión agradable y funcional. Por último, la tecla F5 (VISUALIZAR), muestra en la pantalla o en la impresora, según la opción elegida, el resultado de todo el trabajo.

En resumen, nos hallamos ante un tratamiento de textos cuya utilidad y aplicación están fuera de toda duda.

Puntuación:
Presentación: 9
Claridad: 8
Rapidez: 8
Adicción:



Programa: Idea-base
Tipo: Aplicación
Distribuidor: Idealogic
Formato: Cartucho ROM

Esta base de datos, es una potente herramienta de trabajo, imprescindible desde todos los puntos de vista, ya sea para ser usada con fines didácticos o como una completa base de datos, sencilla y fácil de utilizar.

Es el complemento ideal para los usuarios de los ordenadores MSX de la I Generación, cuyos ordenadores posean dos buses de expansión, ya que, en uno de ellos se conecta el programa y en el otro se podrá conectar la unidad de discos. Esto es una ventaja adicional, al poder usar la unidad de discos conjuntamente con un cassette, convierten al MSX en un completo centro de almacenamiento y tratamiento de información, con la posibilidad de compaginar indistintamente ambos periféricos, además de poder usar conjuntamente otros programas de esta misma casa preparados para tal fin, como por ejemplo, Idea-text y Dim-Calc. Otra de las grandes ventajas de este programa, es que viene en cartucho. Lo que significa, que por un lado se evitan los problemas de carga que suele ser una constante en este tipo de periféricos y por otra parte, la capacidad interna de almacenamiento de datos es bastante considerable, ¡41367 caracteres!

El programa está organizado por menús jerarquizados, siendo el primero que aparece en la pantalla después de la presentación, el principal. Este permite acceder a todas las diversas partes del programa.

En la zona superior de la pantalla, aparecen unas líneas de información. Los datos que hay en ella, indican el menú u operación en que se encuentra. En la zona inferior de la pantalla hay un espacio en blanco que el programa utiliza para informar al usuario de los posibles errores o para comunicarse con él cuando falten datos y en el centro de la pantalla del menú, aparece la lista de posibles acciones que pueden realizarse en ese momento. Para activar una de ellas, bastará con pulsar su número correspondiente.

El menú principal dispone de 8 opciones, que son:

1. Crear fichas.
2. Entrar fichas nuevas.
3. Seleccionar fichas.
4. Cinta.
5. Ordenar.
6. Impresora.
7. Disco.
8. Cambiar colores.

Con estas opciones, se pueden definir todo tipo de ficheros y con el formato que cada uno desee. Pero antes de empezar el proceso de definición de ficheros, hay que tener claros los conceptos de registros y campos. Sin entrar en muchos detalles, principalmente porque éste no es el objeto de esta sección, un fichero está compuesto por varios campos. Lo que quiere decir que, antes que nada habrá que definir los campos, cuya longitud máxima es de 255 caracteres, tamaño suficiente para cualquier tipo de necesidad.

Resumiendo, Idea-base es la base de datos más sencilla y fácil de utilizar que, hoy por hoy podremos encontrar en el mercado.

Puntuación:
Presentación: 8
Claridad: 8
Rapidez: 9
Adicción:

MSX SERVICIO



Núm. 1
¿Qué es el MSX? Su BASIC, periféricos, programas, software.



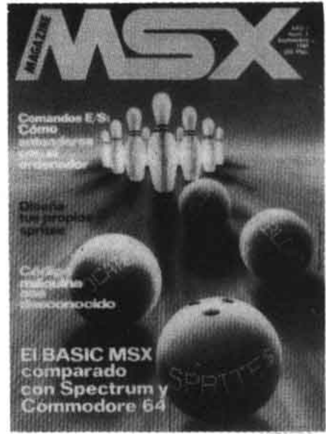
Núm. 2
Generación de sonido. MSX-DOS, el ordenador por dentro, programas, noticias.



Núm. 3
Los joysticks, 256 caracteres programables, Z80 corazón de león, compro/vendo/cambio.



Núm. 4
Las comunicaciones entre ordenadores, la jerga informática, trucos,



Núm. 5
Comandos de entrada/salida, el BASIC MSX comparado con Spectrum y Commodore 64. Código Máquina.



Núm. 6
Los 8 magníficos (test gigante), el bus de expansión, los misterios de la grabación, programas.



Núm. 7
Analizamos el Generador de Sonido. Aplicaciones matemáticas con el ordenador.



Núm. 8
Compact Disc. el periférico del futuro. Test: Dynadata DPC-200. Continuamos con la memoria de vídeo. Libros,



Núm. 9
Características técnicas del Compact Disc. Tratamiento de datos. Test: Quick Disk. Trucos, libros, noticias,



Núm. 10
Características de la II Generación. Los secretos del modo Screen 2. Test: los plotters. Aplicaciones: matrices y determinantes.



Núm. 11
LOGO, un lenguaje educativo. Screen 3: el modo multicolor. Aplicaciones: sistemas de ecuaciones. BASIC para principiantes. Test: Seikosha SP-1000MX.



Núm. 12
SVI-328: precursor del estándar. Aplicaciones: sistemas de ecuaciones II. Código Máquina. Test: Toshiba HX-20.

DE EJEMPLARES ATRASADOS

ESTOS SON LOS EJEMPLARES DE MSX MAGAZINE APARECIDOS EN EL MERCADO CON UN RESUMEN DE SU CONTENIDO



Núm. 13
VG-8235, la I generación en marcha. SVI-318/328: análisis interno. Test: Yamaha CX5M y CX5M II. BASIC: las variables alfanuméricas. Las matemáticas y el ordenador.



Núm. 14
Controle sus errores de programación. Aplicaciones matemáticas: interpolación. Memoria de Vídeo: los sprites. Código Máquina: los registros dobles.



Núm. 15
¿Porqué es lento el BASIC? El procesador de vídeo del SVI-318/328. Test: Sony HB-500P. BASIC: los diagramas de flujo. Los modos de pantalla.



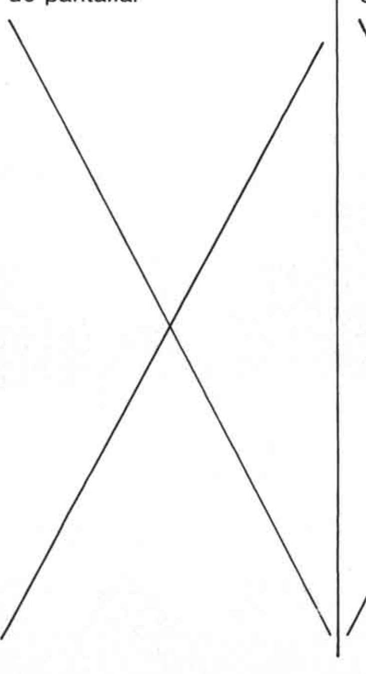
Núm. 16
Dos gigantes frente a frente. Test: VC-10, un osciloscopio muy especial. Síntesis de voz. Utilidades de la RAM. Memoria de vídeo: instrucciones VPEED y VPOKE.

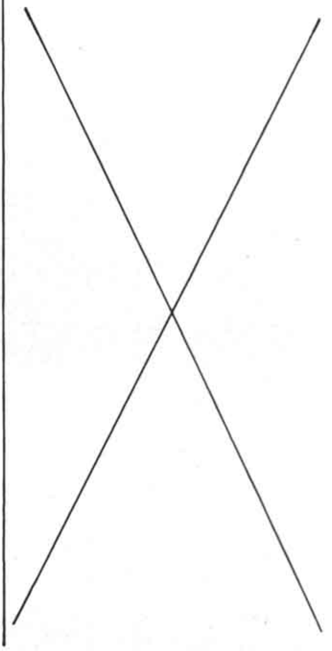


Núm. 17
Robots, trabajadores infatigables. Cómo ahorrar memoria. Test: Mitsubishi ML-G1 y ML-G3. Instrucciones ocultas del Z-80. El procesador de vído del SVI-318/328. Desensamblador.



Núm. 18
Los diskettes al descubierto. El BIOS de la memoria de vídeo. Test: interface RS-232C. Unidad de discos ML-F30D. Utilización de ficheros. SVI-318/328, SCREEN 2.





PARA HACER SU PEDIDO, RELLENE ESTE CUPON, HOY MISMO Y ENVIELO A MSX MAGAZINE
BRAVO MURILLO, 377. Tel. 7337969 - 28020 MADRID

Ruego me envíen los siguientes números atrasados de MSX ____________________
al precio de **300** ptas. cada uno. Cuyo importe abonaré:
☐ POR CHEQUE ☐ CONTRA REEMBOLSO ☐ CON MI TARJETA DE CREDITO
☐ AMERICAN EXPRESS ☐ VISA ☐ INTERBANK

Número de mi tarjeta ____________________
Fecha de caducidad ____________________
NOMBRE ____________________
DIRECCION ____________________
POBLACION ____________________ C.P. ____________
PROVINCIA ____________________

SOFTWARE

Programa: MS-Text
Tipo: Aplicación
Distribuidor: Philips
Formato: Disco/ Cassette

Dentro de los distintos programas de aplicación que hay en el mercado para los MSX, podemos señalar que el MS-Text de Philips ocupa un lugar muy destacado, al ser un programa que no sólo funciona con ordenadores de la I generación, sino que también lo hace en algunos de la II, lo cual es un punto a su favor. Y esto tenemos que resaltarlo, puesto que otros programas con características similares no ejecutaban en otras máquinas, perdiendo una de las más importantes cualidades de los MSX, la compatibilidad.

MS-Text, viene con una presentación muy buena y cuidada, con un manual completo aunque se podría haber entrado en más detalle acerca del programa y sus posibilidades. Del programa, tenemos que resaltar que viene en dos formatos, cinta y disco de 3.5 pulgadas, lo que permite ser empleado por aquellos usuarios que aún no posean la unidad de discos, periférico que lentamente se está haciendo indispensable, sobretodo si queremos que nuestro ordenador se vea potenciado en gran medida. Como es lógico, la diferencia entre la velocidad de carga de uno y otro periférico es abismal, tardando la versión en cinta unos 5 minutos, mientras que en disco apenas llega a 30 segundos. Es importante destacar esta característica, ya que ninguna casa prepara sus programas de gestión (la que los tenga) en dos formatos tan populares.

Volviendo al programa, este puede ser empleado por todo tipo de usuarios, desde los más pequeños hasta los profesionales, aunque para estos últimos haga falta otro que lo complemente como es el caso de MS-Base, una de las bases de datos más potentes que hay aunque presenta problemas de compatibilidad. Estos dos programas tienen estructura interactiva, lo que significa que se pueden utilizar los datos de uno (la base de datos) con el otro (el tratamiento de textos) para realizar mailings, informes, entre otras cosas.

Respecto al programa, la pantalla principal muestra en la línea superior el estado actual en que nos hallamos, a la vez que nos va mostrando la capacidad de memoria con que contamos. Empezamos con espacio para más de 31.000 caracteres, tamaño suficiente para realizar informes, cartas, artículos, etc., y disponemos de 8 opciones que van desde la de crear texto hasta controlar los colores de la pantalla, pudiendo elegir cualquier opción. Todas ellas vienen bien explicadas en el manual, sin embargo, no habrá que utilizarlo con frecuencia al ser estas bastante claras. De todas las opciones que tiene el programa, la que más nos gustó fue la de visualizar el texto por pantalla, ya que permite ver con detalle cómo va a quedar el texto en la impresora, además de facilitar la apertura de ventanas en el texto, y muchas cosas más.

Estamos ante el programa ideal para todos aquellos que necesiten un tratamiento de textos útil, versátil y potente.

Puntuación:
Presentación: 10
Claridad: 8
Rapidez: 9
Adicción:

Programa: dBASE II
Tipo: Aplicación
Distribuidor: Philips
Formato: Disco 3.5

¡Por fin! ya tenemos un programa de aplicación en condiciones, con las mismas funciones y características que el dBASE II del IBM PC. Este programa, es un sistema de manejo de datos potente y flexible, con un lenguaje de programación propio que potencia, aún más si cabe, esta base de datos.

dBASE II (para los IBM PC) se empezó a comercializar en 1981, lo que significa que ya lleva bastante tiempo en el mercado (actualmente existe el dBASE III para PC, siendo éste una versión actualizada del dBASE II). Aún así, la versión existente para MSX, a pesar de ser la anterior sigue siendo lo suficientemente actual y potente como para no perder vigencia frente a otros programas, y es más, tiene unas características poco usuales en un programa de aplicación como veremos más adelante. De entrada, el programa está preparado para que funcione tanto en los ordenadores de la I como de la II generación.

El manual que acompaña al programa es amplio y completo, explicando detalladamente todas y cada una de las cualidades del dBASE II, así como el lenguaje de programación que completa el manejo de este programa. Al final existe un ejemplo orientativo. Veremos algunas de las muchas instrucciones que posee.

Para empezar, el comando *Help,* le permitirá acceder al texto explicativo de todas las instrucciones y funciones que componen el programa. La lista, a primera vista interminable, está muy completa pero a pesar de ello, no implica que sea complicado de manejar el programa, es más, una persona que nunca haya tenido contacto alguno con este tipo de aplicaciones podrá desenvolverse bien frente al maremagnum de instrucciones y comandos que hay. También cabe destacar la cualidad que tiene este programa de que no funciona con menús, sino con instrucciones directas, lo que significa que el usuario tendrá que memorizarse las instrucciones existentes o mirar a cada rato la lista de comandos. En realidad, esto tiene una ventaja bastante importante, debido a que los programas que funcionan mediante menús tienden a ser pesados una vez se sepan manejar a fondo, esto se elimina con el dBASE II cuyas instrucciones acceden directamente a la operación a realizar sin tener que pasar por menú alguno. De cualquier manera, el dBASE II está planteado en base a que el usuario novato lo es sólo una vez, ya que con el uso y abuso del programa se va adquiriendo una experiencia necesaria e imprescindible en su manejo. Con ello se consigue transformar los datos, en información muy útil, ya sea agrupándola en un conjunto de información como bloque o de cualquier otra manera.

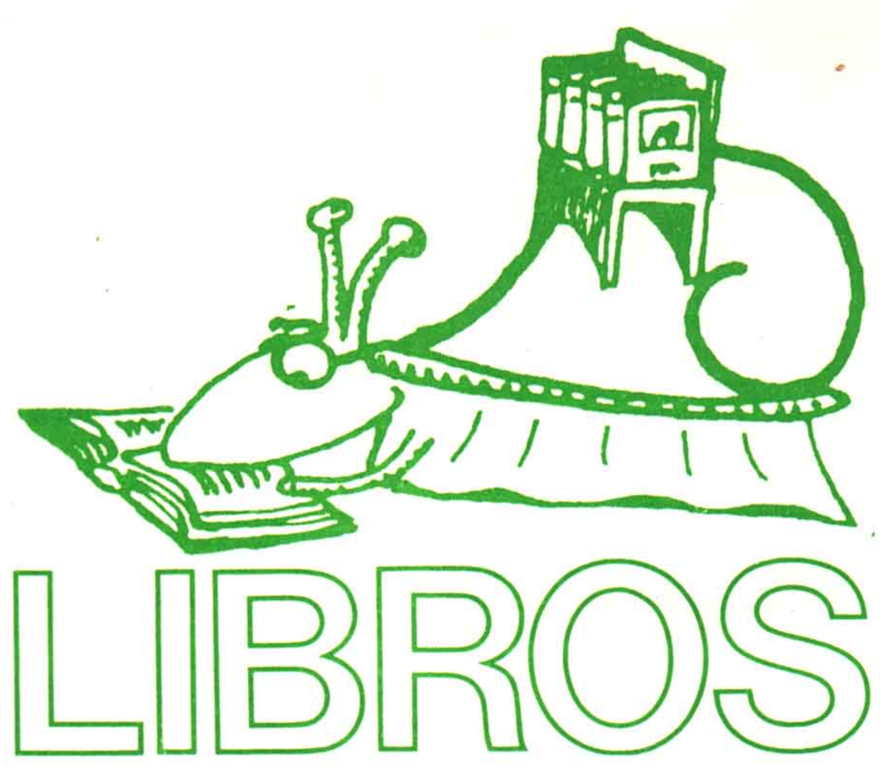
Este primer contacto con el dBASE II, nos ha servido para mostrar el inmediato futuro que podemos esperar de los MSX cuando se posee un programa en condiciones para trabajar y manifiesta una de las posibles tendencias de los MSX de la II generación, ordenadores personales para la pequeña empresa, algo totalmentne factible cuando se poseen unos medios como estos.

De cualquier manera, este programa merece la pena ser comentado mucho más extensamente de lo que permite el espacio dedicado a esta sección, por lo que en próximos números dedicaremos un artículo a desvelar los misterios del dBASE II.

```
> ?       - muestra una lista de expresiones
> ??      - muestra una lista de expresiones sin saltar a la linea siguiente
> @       - muestra datos formateados por usuario en pantalla o impresora
> ACCEPT - permite entrar cadenas de caracteres en variables de memoria
> APPEND - incorpora ficheros dBASE o ficheros en System Data Format (SDF) o
              ficheros delimitados
> BROWSE - muestra o edita datos usando la pantalla completa
> CANCEL - cancela la ejecucion de un fichero de mandatos
> CASE - una posibilidad de ejecucion mas en una estructura DO CASE
> CHANGE  - edita columnas o campos
> CLEAR - cierra los archivos y borra todas las variables de memoria
> CONTINUE - continua la ejecucion de un mandato LOCATE
> COPY  - copia bases de datos existentes para crear duplicados
> COUNT - cuenta el numero de registros siguiendo un cierto criterio
> CREATE - crea una nueva estructura de fichero de base de datos
> DELETE - borra ficheros y marca registros para ser borrados
> DISPLAY - muestra registros, campos y expresiones
> DO     - permite la ejecucion de ficheros de mandatos y permite bucles
              estructurados en ficheros de madatos
> EDIT  - altera campos de datos especificos en una base de datos

A la espera
```

Puntuación:
Presentación: 10
Claridad: 9
Rapidez: 9
Adicción:



Libro: FORTH. Anatomía de un lenguaje inteligente
Editorial: INGELEK, S.A.
Páginas: 109
Precio: 395 ptas. (IVA incluido)

Hay quienes, al acercarse por primera vez al FORTH, se sienten extrañados de sus peculiaridades y un poco perdidos, llegando incluso a tacharlo de *«raro»* y a abandonar cualquier intento de profundizar en él. Sin embargo, el FORTH es un lenguaje moderno con el que es natural la programación *«de abajo a arriba»,* que permite obtener el máximo provecho del ordenador y disponer, desde un lenguaje de alto nivel, de la rapidez y potencia del lenguaje máquina.

Así pues, el tema central de este libro es el estilo característico de programación del FORTH, su filosofía y lo que se podrá o no hacer con él.

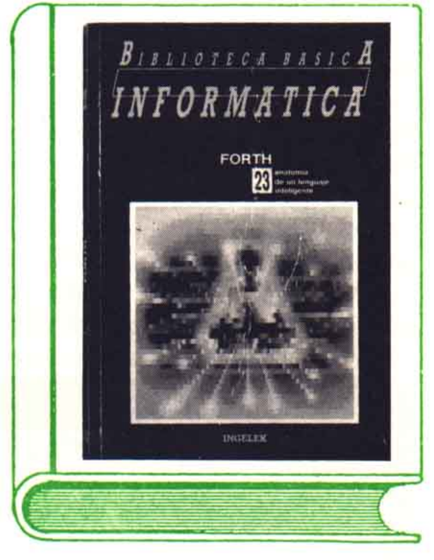
Por supuesto, siempre resulta interesante conocer la historia de un lenguaje para poder comprenderlo.

El primer capítulo del libro está dedicado a un somero repaso del origen del FORTH, la figura de sus creadores, y las aplicaciones hacia las cuales se orientó en sus orígenes.

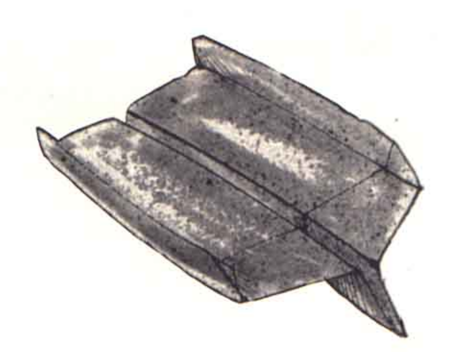
Para que se hagan una idea de la potencia de este lenguaje, la sociedad compuesta por sus creadores Moore y Rather se dedicó en sus comienzos a aplicaciones tales como el control y tratamiento de datos recibidos por un radiotelescopio.

La base del FORTH es la «palabra» (en inglés, *word*), que expresa un concepto similar al procedimiento en Pascal, pero con algunos matices diferentes. De hecho, el concepto «palabra» en FORTH admite una mayor flexibilidad y potencia. Al análisis de este tema se dedica el capítulo segundo.

Otra característica notable de este lenguaje es el uso de la notación RPN (*Reverse Polish Notation:* Notación Polaca Inversa), con la cual estarán familiarizados los lectores que ya hayan manejado ciertos modelos de calculadoras Hewlett-Packard. Se trata de un sistema basado en una Pila o *Stack,* que ofrece una mayor rapidez y menor gasto de memoria para los cálculos matemáticos. Los siguientes capítulos explican esta notación, así como las operaciones aritméticas y el manejo del *Stack.*



La flexibilidad del FORTH nace de la posibilidad de crear, redefinir y eliminar palabras (las palabras representan acciones o grupos de



acciones) y trabajar con un «diccionario» de palabras. De este modo, partiendo de unas pocas palabras básicas, el programador puede crear su propio diccionario, adaptando así el lenguaje a sus necesidades. El proceso de creación, redefinición y destrucción de palabras se contempla en los capítulos 6 y 7.

No obstante, no todo son ventajas en el FORTH. La flexibilidad y la rapidez de ejecución tienen un precio: la limitación del uso de matemáticas en coma fija. Para ver más detalles léase el capítulo 8.

A continuación se describen las estructuras de Forth que tienen paralelismo en otros lenguajes: bucles iterativos *DO...LOP,* bucles condicionales *DO...UNTIL, IF...THEN...ELSE,* etc., y definición y manejo de variables, constantes y tablas (matrices o arrays).

Por último, aprendemos la forma de mantener un diálogo interactivo con el FORTH, utilizando las funciones de E/S (entrada/salida), así como la posibilidad de manejar diversas bases de numeración. Al final del libro, como siempre, encontramos dos apéndices muy útiles, uno con las palabras básicas del FORTH, y otro con el código ASCII.

En resumen, se trata de un libro enfocado hacia aquellos que desean introducirse en el siempre intrincado mundo de un nuevo lenguaje partiendo desde su base. No se trata, desde luego, de un libro avanzado destinado a los que ya conocen el FORTH.

Título: MSX Lenguaje Máquina
Autor: Dullin - Strassenburg
Editorial: Ferre Moret S.A.
Páginas: 305

Dado el elevado número de ventas de ordenadores personales, son cada día más las personas interesadas en los temas referentes a estos aparatos, y entre ellos el Lenguaje Máquina. Sin embargo, este lenguaje presenta una cierta complejidad, y no es sencillo encontrar el libro adecuado para aprender partiendo desde un nivel «cero», es decir, para el que no conoce nada de dicho lenguaje.

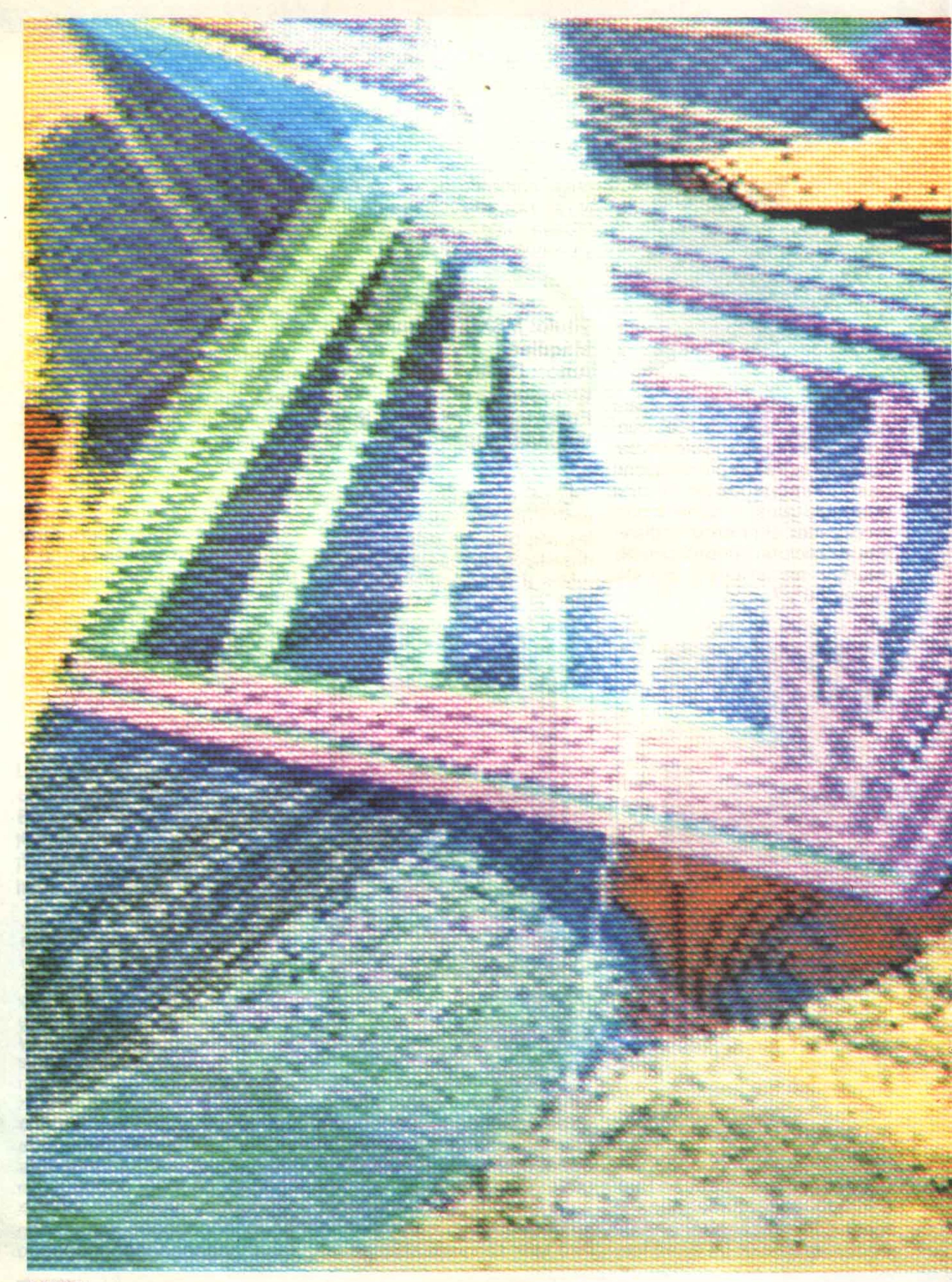


Por ello, los usuarios de MSX están de enhorabuena, ya que el libro MSX Lenguaje Máquina les brinda la oportunidad de introducirse en este campo de forma sencilla a través de sus páginas, en las que abundan los ejemplos. Además, los autores parten de la base de que el lector es totalmente nóvel en la informática, con lo cual se han preocupado de preparar un capítulo introductorio al ordenador, la *CPU,* los sistemas de numeración y la aritmética binaria, de modo que el lector poco experimentado en este terreno no se encuentre perdido en su recorrido por el libro.

El lector no debe pensar que leyendo este libro ya sabrá realizar programas profesionales en código máquina. Lo único que pretende el libro es enseñar el lenguaje de la *CPU,* cómo se estructura y cómo se utiliza el lenguaje ensamblador para programar; esto es, las técnicas fundamentales de programación. El lector necesitará practicar mucho para conseguir un buen nivel de conocimiento de este lenguaje.

Hay que destacar que los autores se han preocupado de proporcionar al lector las herramientas mínimas para que pueda practicar desarrollando los ejemplos. Así, se incluyen los listados de un programa ensamblador de prácticas, desarrollado en *BASIC,* y de un monitor «paso a paso», también desarrollado en *BASIC.* Con estos dos programas podrá ejercitarse y verificar el funcionamiento de las rutinas-ejemplo, aprendiendo así a elegir en cada caso la instrucción del *Z80* necesaria.

Al final del libro se incluyen un apéndice con las rutinas del sistema operativo en *ROM* más útile, así como las tablas de instrucciones del microprocesador *Z80* que facilita la compañía Zilog, fabricante de dicha *CPU.*





Diseño de Sprites

Este programa permite la creación de *sprites,* almacenamiento en disco, y pruebas de animación con *sprites,* en todos los modos, para los MSX de segunda generaciódn. El programa original está escrito en un PHILIPS VG-8235.

Explicamos a continuación, por encima, el listado del programa.

Líneas 10-160. Pantalla de presentación e inicialización de variables. Cambios en las teclas de función.

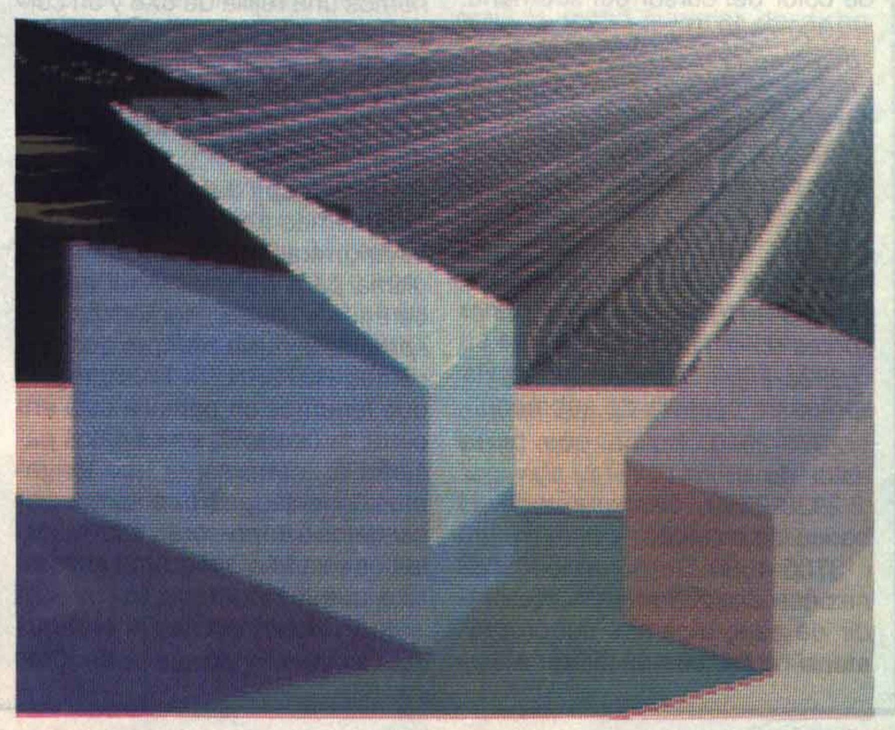
170-230. Diseño de un *sprite* para el cursor. Este *sprite* puede ser modificado más adelante por vosotros.

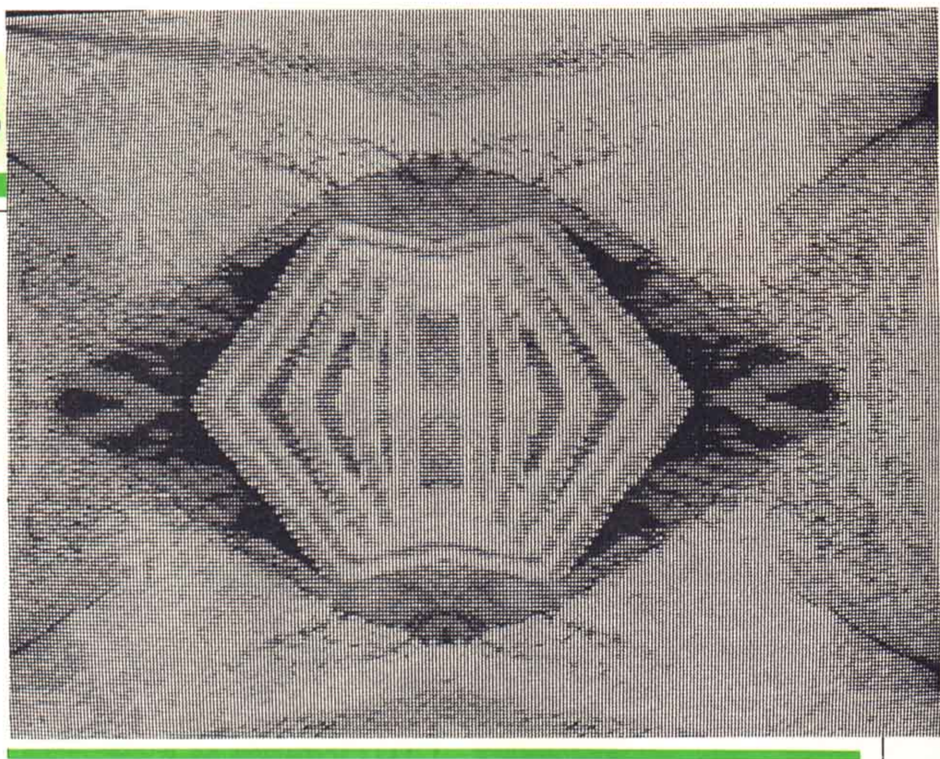
250-430. Bucle del menú principal.

450-530. Rutina para la creación de un *sprite* aleatorio.

540-900. Creación pantalla principal. La presentación del programa es una de sus principales características.

910-920. Rutina para encender/apagar celdillas.





930-1090. Rutina para convertir el *sprite* seleccionado a datos numéricos.

1100-1220. Dos rutinas de impresión en la ventana de menús.

1230-1240. Apaga/enciende impresora de pantalla.

1250-1330. Rutina de inversión de *sprite*.

1340-1510. Rutina para efecto espejo vertical.

1520-1640. Rutina borrado de rejilla.

1650-1760. Rutina giro de *sprite*.

1770-1870. Rutina efecto espejo en horizontal.

1880-2120. Rutina de almacén para *sprites*.

2130-2190. Rutina de volcado de todos los *sprites* a impresora. Esta rutina va leyendo *sprite* a *sprite* de tal manera que si encuentra uno vacío lo ignora y sigue a por otro.

2200-2360. Rutina de conversión de *sprites* a datos numéricos en submenú.

2370-2380. Rutina de cambio de color del cursor del submenú.

2390-2480. Rutina para guardar *sprites* en la memoria.

2490-2520. Desplazamiento cursor submenú.

2530-2650. Rutina que saca *sprite* de memoria a rejilla de diseño.

2660-2750. Rutinas de movimiento del cursor del submenú.

2760-2810. Rutina para poner *sprites* en el submenú.

2820-2860. Rutina que quita *sprites* en pantalla de submenú.

2870-2910. Rutina que borra datos de *sprite*.

2920-2690. Gestiona todo el banco de pruebas.

3700-4880. Gestiona operaciones con disco. Podemos prescindir de este, digamos subprograma, si no pensamos utilizar almacenamiento en disco eliminando la línea 2020 y las líneas 4890 a 5020.

4890-5020. Rutina de detección de errores en disco.

5030-5460. Rutina de animación de *sprites*.

El programa se compone básicamente de cuatro pantallas o menús. El primero de todos (el menú principal) permite la creación y modificación de sprites. Tenemos una rejilla de 8X8 y un cursor parpadeante en forma de cruz. Este se mueve con los cursores y para dibujar o borrar una celda basta con pulsar la barra espaciadora.

A la derecha de la rejilla de diseño tenemos tres zonas, una para binario, otra para hexadecimal y la última para decimal. Aquí aparecerán los datos numéricos del *sprite* de la rejilla siempre que pulsemos RETURN. Si la impresora que tenemos en pantalla está encendida (tecla F7) estos datos pasarán a la impresora que tengamos conectada. Si intentamos pasar datos a impresora y esta no está conectada el programa esperará a que lo hagamos.

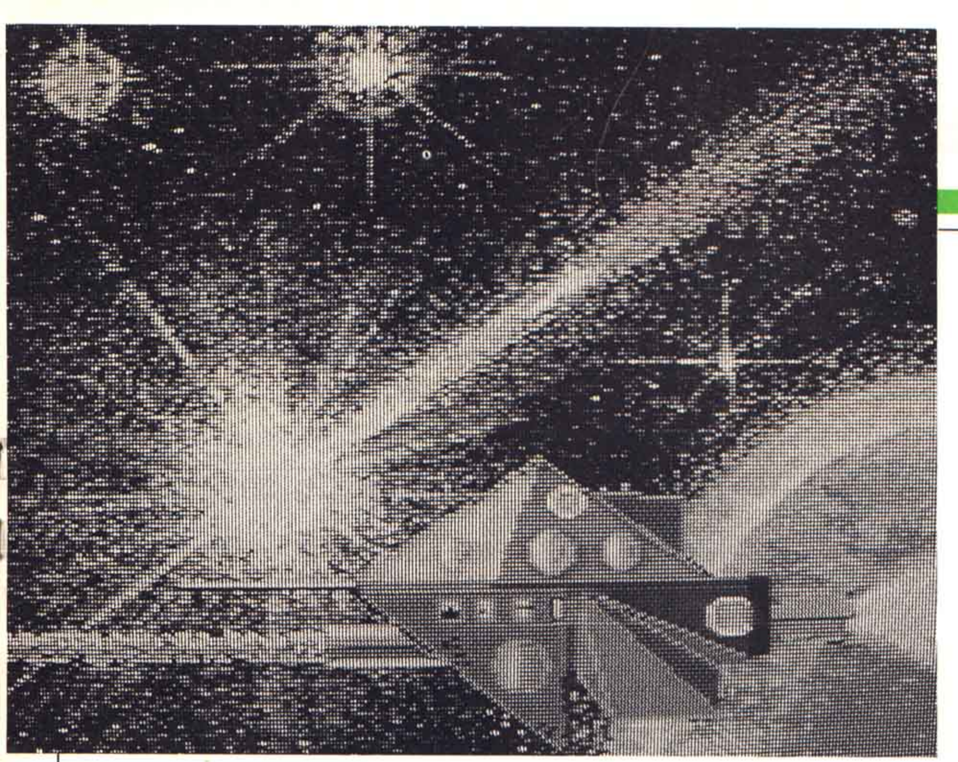
Justo abajo de la rejilla tenemos dos instrucciones. La tecla CLR limpia la rejilla, borra los datos numéricos y el *sprite* en tamaño natural que se ha ido dibujando en un cuadradito blanco que hay más abajo.

ESC pasa a impresora (si está conectada) todos los *sprites* que tengamos en memoria.

La tecla de función F10 comienza de nuevo el programa borrando todos los datos que tengamos hasta el momento.

Nos queda por ver las teclas F1 a F6. Todas menos la F5 alteran de alguna forma nuestro diseño. La F1 invierte las celdillas una por una, es decir, la que estaba encendida se apaga y viceversa. La F2 hace un efecto de espejo en sentido horizontal. La F3 gira noventa grados el *sprite* de la rejilla. La F4 realiza la misma función que la F2 pero en sentido vertical. La F6 va encendiendo o apagando celdillas aleatoriamente. La F5 cambia el menú de la pantalla principal.

Cuando pulsamos esta tecla las F1 a F5 cambian de significado, el cursor de la rejilla se vuelve inactivo y aparece otro cursor dentro de esta ventana. Nos hallamos en el almacén de *sprites*. Con los cursores nos podemos desplazar por



256 posiciones de memoria a nuestro capricho.

La tecla F1 guarda el *sprite* dibujado en la rejilla en la posición donde se halle el nuevo cursor. Al mismo tiempo este *sprite* queda almacenado en la memoria del ordenador.

La F4 lleva un *sprite* del almacén a la rejilla para su posible modificación. F5 devuelve el control al menú anterior. F2 nos lleva a un submenú y pantalla diferente. Aquí podemos salvar, cargar, listar y borrar en disco los ficheros de *sprites.*

La F3 nos conduce a una pantalla particularmente útil donde se pueden variar el color de *sprites,* moverlos, colores de fondo, superponerlos, probar animación, etc.

Cada una de estas pantallas, disco y banco de pruebas se explican a continuación.

La pantalla dedicada al disco presenta claramente el menú de opciones así como la hora del reloj interno. CARGAR, LISTAR y BORRAR no presenta ningún problema. Pero antes de meternos en la opción SALVAR debemos indicar con los cursores y la barra espaciadora de cuál a cuál queremos grabar en disco. Podemos tener 30 *sprites* almacenados e interesarnos grabar sólo del 1 al 20. De todas formas antes de grabar el fichero se pregunta por esta cuestión.

Otra cosa a tener en cuenta es que no es necesario teclear la extensión, SPR en este proceso. El programa lo inserta automáticamente.

Los posibles mensajes de error de disco están indicados en una subrutina de errores con lo cual el programa no pierde el control (discos protegidos, falta de disco en la unidad, etc.).

La opción PRUEBAS es otro submenú del programa. Aquí nos encontramos con una gran ventana dentro de la cual se ubican todos los *sprites* diseñados o cargados desde un disco. Tenemos también la hora del reloj interno. Pasamos ahora a explicar cada tecla.

F1 y F2 seleccionan un *sprite* dentro del rango de SELECT (cuando entramos por primera vez de 0 al 31), el *sprite* seleccionado parpadea. Con los cursores podemos moverlo dentro de la ventana. Cada *sprite* se desplaza por una determinada superficie, así el *sprite* o (una cruz) pasará por delante del 1 y este por delante del 2 y así sucesivamente.

F3 cambia el color del *sprite* seleccionado desde el negro (O) hasta el blanco (15) para empezar de nuevo con el negro.

F4 Varía el color de fondo de la ventana de igual modo que F3.

F5 Devuelve el control al menú principal.

La tecla BS accede a un submenú para animación de *sprites.* Este submenú pregunta de cuál a cuál sprite se quiere comprobar la animación. Los cursores arriba/abajo aumentan o disminuyen la carencia de animación y BS regresa de nuevo al menú. Para usar adecuadamente esta opción hay que diseñar lógicamente los *sprites* adecuados y en el orden correcto.

ESC cambia el modo de *sprite* (8X8 normal, 8X8 ampliado, 16X16 normal y 16X16 ampliado). Cuando queramos hacer *sprites* de 16X16 debemos tener en cuenta la organización de estos en memoria, por ejemplo, el *sprite* 0 y 2 del almacén en modo 16x16 sería la mitad superior de dicho *sprite* y el 1 y el 3 la inferior. Observamos que van de cuatro en cuatro (0/3, 4/7, 8/11, etc.). Es cuestión de practicar un poco.

Las funciones de las teclas ESC y SELECT están programadas de tal manera que no se pueda poner por ejemplo *sprites* 127 al 159 en modo 16X16 ya que en este tipo de sprite sólo hay hasta 60.

Sólo resta repetir que este programa, aunque largo merece la pena, está pensado para MSX de segunda generación ya que principalmente utiliza la SCREEN 5, mucho más rápida y amplia que la SCREEN 2.

Os deseamos felices diseños.

José Carlos Tomás

Una observación

El programa, por lo extenso que es (unos 19K), lo hemos reducido y debido a esto, algunas líneas resultan algo difícil de leer. Las más problemáticas son la 160 y el bloque formado por las líneas 320 a 420, donde los caracteres especiales, pequeños de por sí, son algo borrosos. Por este motivo, a continuación reproducimos dichas líneas.

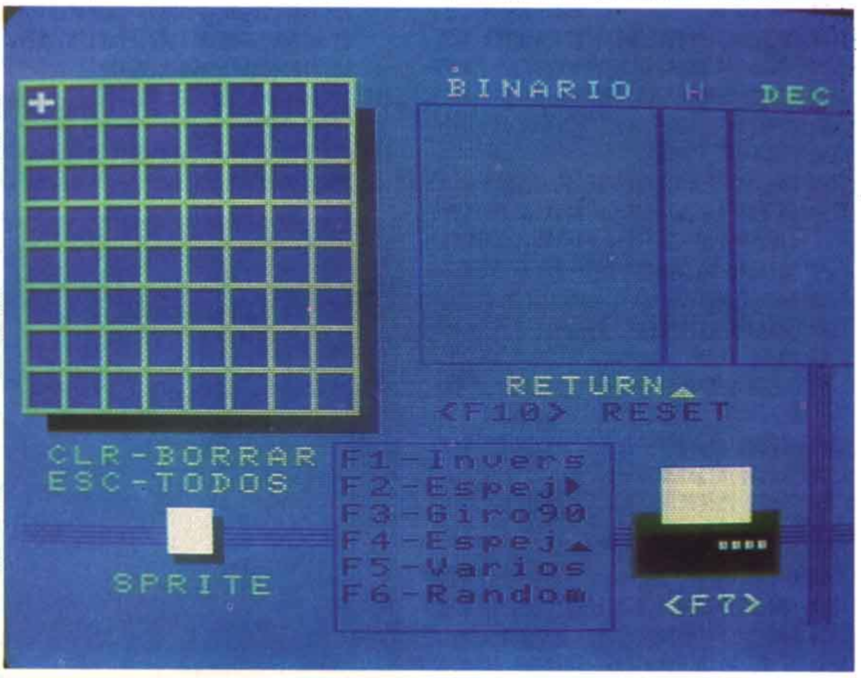
Para finalizar, tenemos que resaltar que, aunque el programa se haya desarrollado en un Philips VG-8235, funcionará con cualquier ordenador MSX de la II generación. En caso de intentar ejecutarlo en un ordenador de la I generación, aparecerá un mensaje de error, siendo imposible ejecutarlo.

```
160  KEY OFF:KEY 1,"½":KEY 6,"¼":KEY
     2,"¾":KEY 3,"n":KEY 4,"%":KEY 5,
     "r":KEY 7,"r":KEY 10,"∞"

320  IF E$="∞" THEN RUN
330  IF E$=CHR$(13) THEN GOSUB 2870:G
     OSUB 930:BEEP
340  IF E$="r" THEN GOSUB 1230
350  IF E$=CHR$(11) THEN GOSUB 1520:G
     OSUB 2870:LINE(43,165)-(55,181),
     15,BF
360  IF E$=CHR$(27) THEN GOSUB 2130
370  IF E$="½" THEN GOSUB 1250
380  IF E$="¾" THEN GOSUB 1340
390  IF E$="n" THEN GOSUB 1650
400  IF E$="%" THEN GOSUB 1770
410  IF E$="r" THEN GOSUB 1880
420  IF E$="¼" THEN GOSUB 440
```

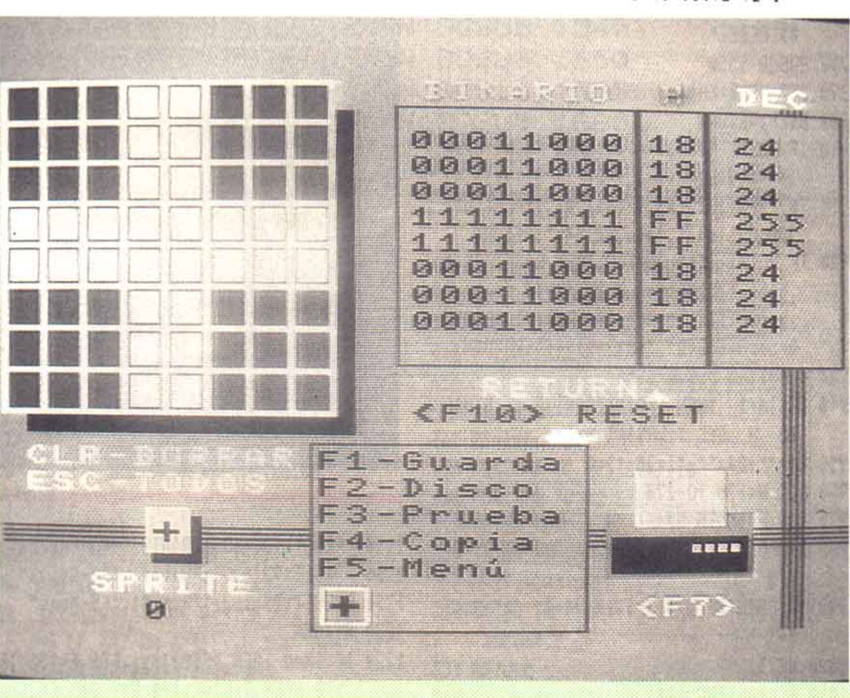
```
10 SCREEN 5:WIDTH 37:COLOR 7,1,1:CLS
20 OPEN"grp:"AS#1
30  X=3:Y=4:SP=1:CC=14:B1=0:DIM SP$(8),S
    T%(255):DC=30720:N$=SPACE$(12)
40  R=0:I=0:P3=0:X2=93:DIM CO%(255),X%(2
    56),Y%(256):DI=30720
50 PRESET(50,52):PRINT#1,"┌──────
    ──────────┐"
60 PRESET(50,58):PRINT#1,"|DISEÑO DE SP
    RITES|"
70 PRESET(50,66):PRINT#1,"└──────
    ──────────┘"
80 PRESET(50,100):PRINT#1,"┌─────
    ───────────┐"
90 PRESET(50,108):PRINT#1,"|ESPERA UN M
    OMENTO|"
100 PRESET(50,116):PRINT#1,"└─────
    ───────────┘"
110 COLOR 11:PRESET(10,180):PRINT#1,"◊JO
    SE C.TOMAS◊"
120 COLOR 5:PRESET(10,190):PRINT#1,"◊MSX
    2◊6/10/1986◊
130 LINE(0,80)-(255,90),6,BF
140 LINE(0,83)-(255,87),8,BF
150 LINE(0,85)-(255,86),9,BF:FOR F%=0 TO
     255:X%(F%)=INT(RND(-TIME)*240):Y%(
    F%)=INT(RND(-TIME)*100):CO%(F%)=1:L
    INE(F%+1,80)-(F%+1,90),7:LINE(F%,80
    )-(F%,90),1:NEXT:BEEP
160  KEY OFF:KEY 1,"½":KEY 6,"¼":KEY 2,"
    ¾":KEY 3,"n":KEY 4,"%":KEY 5,"r":KE
    Y 7,"r":KEY 10,"•"
170 SCREEN 5,0,0
180  FOR F=1 TO 8
190  READ A$
200  S$=S$+CHR$(VAL("&h"+A$))
210  NEXT
220  SPRITE$ (0)=S$
230  DATA 18,18,18,ff,ff,18,18,18
240 GOSUB 540
250  E$=INKEY$
260  PUT SPRITE 0,(X,Y),15,0
270  A=STICK(0)
280  X=X-12*((A=2 OR A=3 OR A=4))+12*((A
     =6 OR A=7 OR A=8)):IF X<0 THEN X=87
     :ELSE IF X=>96 THEN X=3
290  Y=Y-16*((A=4 OR A=5 OR A=6))+16*((A
     =8 OR A=1 OR A=2)):IF Y<3 THEN Y=11
     6:ELSE IF Y=>119 THEN Y=4
300  PUT SPRITE 0,(X,Y),13,0
310  IF STRIG(0) THEN GOSUB 910
320  IF E$="•" THEN RUN
330  IF E$=CHR$(13) THEN GOSUB 2870:GOSU
     B 930:BEEP
340  IF E$="r" THEN GOSUB 1230
350  IF E$=CHR$(11) THEN GOSUB 1520:GOSU
     B 2870:LINE(43,165)-(55,181),15,BF
360  IF E$=CHR$(27) THEN GOSUB 2130
370  IF E$="½" THEN GOSUB 1250
380  IF E$="¾" THEN GOSUB 1340
390  IF E$="n" THEN GOSUB 1650
400  IF E$="%" THEN GOSUB 1770
410  IF E$="r" THEN GOSUB 1880
420  IF E$="¼" THEN GOSUB 440
430  GOTO 250
440 ' sprite aleatorio
450 COLOR 15:PRESET(93,193):PRINT#1,"F6-
     Random"
460 RD=INT(RND(TIME)*13)
470 FOR Y=4 TO 128 STEP 16
480 FOR X=3 TO 96 STEP 12
490 RN=INT(RND(TIME)*15)
500 IF RN>RD THEN GOSUB 910
510 NEXT:NEXT:BEEP
520 COLOR 1:PRESET(93,193):PRINT#1,"F6-R
     andom"
530 RETURN
540 COLOR 10,5,5:CLS:IM=0
550 DEF USR0=&H41:U=USR0(0)
560 FOR F=172 TO 178 STEP 2:LINE(0,F)-(2
     55,F),4:NEXT
570 FOR F=230 TO 236 STEP 2:LINE(F,0)-(F
     ,10),4:NEXT
580 FOR F=230 TO 236 STEP 2:LINE(F,111)-
     (F,210),4:NEXT
590 LINE(8,12)-(102,136),1,BF
600 LINE(0,0)-(96,128),4,BF
610 LINE(45,168)-(58,185),1,BF
620 LINE(43,165)-(55,181),15,BF
630 LINE(115,10)-(185,110),4,B
640 LINE(186,10)-(206,110),4,B
650 LINE(207,10)-(249,110),4,B
660 LINE(176,166)-(220,190),12,BF:LINE(1
```

```
78,175)-(218,188),1,BF:FOR F=202 TO 215 STEP 4:LINE(F,178)-(F+2,180),14,BF:NEXT
670 LINE(185,150)-(211,170),11,BF:COLOR 10:PRESET(186,198):PRINT#1,"<F7>"
680 LINE(90,140)-(170,210),4,B
690 LINE(91,141)-(169,209),5,BF
700 GOSUB 1100
710 COLOR 14:PRESET (124,0):PRINT#1,"BINARIO
720 COLOR 9:PRESET (194,0):PRINT#1,"H
730 COLOR 3:PRESET(217,0):PRINT#1,"DEC
740 PRESET(8,142):PRINT#1,"CLR-BORRAR"
750 PRESET(8,152):PRINT#1,"ESC-TODOS"
760 PRESET(27,190):PRINT#1,"SPRITE"
770 PRESET(140,115):PRINT#1,"RETURN▲"
780 COLOR 1:PRESET(120,125):PRINT#1,"<F10> RESET"
790 GOSUB 1550
890 DEF USR0=&H44:U=USR0(0)
900 BEEP:RETURN
910 IF POINT (X+4,Y+4)=4 THEN PAINT (X+4,Y+4),3:LINE(X-1,Y-2)-(X+8,Y+11),4,B:PSET(X/12+45,Y/16+170),1:PLAY"s0m300o8v6l64a","s0m300l64c":RETURN
920 IF POINT (X+4,Y+4)=3 THEN PAINT (X+4,Y+4),4:LINE(X-2,Y-3)-(X+10,Y+13),3,B:PSET(X/12+45,Y/16+170),15:PLAY"s0m30o8v6l64d","s0m300l64f":RETURN
930 ' rutina conversion
940 IF IM=1 THEN LPRINT"SPRITE    BINARIO  H  DEC ":LPRINT
950 A$="":B$="":Y1=20:Y2=0
960 FOR N=170 TO 177
970 FOR F=45 TO 52
980 IF POINT(F,N)=1 THEN A$=A$+"1":B$=B$+"█"
990 IF POINT(F,N)=15 THEN A$=A$+"0":B$=B$+" "
1000 NEXT
1010 BI=VAL("&B"+A$)
1020 H$=HEX$(BI):IF LEN(H$)=1 THEN H$="0"+H$
1030 COLOR 1:PRESET(120,Y1):PRINT#1,A$
1040 COLOR 1:PRESET(190,Y1):PRINT#1,H$:PRESET(208,Y1):PRINT#1,BI:PLAY"o8l16s8m300d"
1050 IF IM=1 THEN LPRINT B$;" ";A$;" ";H$;" ";BI
1060 A$="":B$="":Y1=Y1+10:Y2=Y2+1
1070 NEXT
1080 IF IM=1 THEN LPRINT"-------------------------"
1090 RETURN
1100 COLOR 1:PRESET(93,143):PRINT#1,"F1-Invers"
1110 PRESET(93,153):PRINT#1,"F2-Espej▶"
1120 PRESET(93,163):PRINT#1,"F3-Giro90"
1130 PRESET(93,173):PRINT#1,"F4-Espej▲"
1140 PRESET(93,183):PRINT#1,"F5-Varios"
1150 PRESET(93,193):PRINT#1,"F6-Random"
1160 RETURN
1170 COLOR 1:PRESET(93,143):PRINT#1,"F1-Guarda"
1180 PRESET(93,153):PRINT#1,"F2-Disco"
```




```
1190 PRESET(93,163):PRINT#1,"F3-Prueba"
1200 PRESET(93,173):PRINT#1,"F4-Copia"
1210 PRESET(93,183):PRINT#1,"F5-Menú"
1220 RETURN
1230 IF IM=0 THEN BEEP:IM=1:FOR F=155 TO 170 STEP 2:LINE(190,F)-(205,F),1:NEXT:LINE(182,178)-(184,180),8,BF:RETURN
1240 IF IM=1 THEN BEEP:IM=0:FOR F=155 TO 170 STEP 2:LINE(190,F)-(205,F),11:NEXT:LINE(182,178)-(184,180),1,BF:RETURN
1250 ' Invierte
1260 COLOR 15:PRESET(93,143):PRINT#1,"F1-Invers":BEEP
1270 GOSUB 2870
1280 FOR Y=4 TO 128 STEP 16
1290 FOR X=3 TO 96 STEP 12
1300 GOSUB 910
1310 NEXT:NEXT:BEEP
1320 COLOR 1:PRESET(93,143):PRINT#1,"F1-Invers
1330 RETURN
1340 ' Espejo▶
1350 COLOR 15:PRESET(93,153):PRINT#1,"F2-Espej▶":BEEP
1360 GOSUB 1520:GOSUB 2870
1370 P=3:P1=4
1380 FOR F=170 TO 177
1390 FOR N=52 TO 45 STEP-1
1400 IF POINT(N,F)=1 THEN PAINT(P,P1),3:LINE(P-1,P1-2)-(P+8,P1+11),4,B:PLAY"o3l64s15m200e"
1410 P=P+12:NEXT
1420 P=3:P1=P1+16:NEXT
1430 P=45:P1=170
1440 LINE(43,165)-(55,181),15,BF
1450 FOR F=4 TO 128 STEP 16
1460 FOR N=3 TO 96 STEP 12
1470 IF POINT (N,F)=3 THEN PSET(P,P1),1:PLAY"o8l64s15m300g"
1480 P=P+1:NEXT
1490 P=45:P1=P1+1:NEXT
1500 COLOR 1:PRESET(93,153):PRINT#1,"F2-Espej▶"
1510 BEEP:RETURN
1520 ' Borrado pantalla gráfrica
1530 SOUND6,1:SOUND8,16:SOUND1,13:SOUND7,1:SOUND13,0:SOUND12,25
1540 LINE(0,0)-(97,129),4,BF
1550 FOR Z=0 TO 1
```

```
1560 FOR F=0 TO 96 STEP 12
1570 LINE(F+Z,0)-(F+Z,128),3
1580 NEXT
1590 FOR F=0 TO 128 STEP 16
1600 LINE(0,F+Z)-(96,F+Z),3
1610 NEXT
1620 NEXT
1630 PSET(97,129),3:BEEP
1640 RETURN
1650 ' Giro 90
1660 COLOR 15:PRESET(93,163):PRINT#1,"F3
     -Giro90":BEEP
1670 GOSUB 1520:GOSUB 2870
1680 P=3:P1=4
1690 FOR F=45 TO 52
1700 FOR N=177 TO 170 STEP-1
1710 IF POINT(F,N)=1 THEN PAINT(P,P1),3:
     LINE(P-1,P1-2)-(P+8,P1+11),4,B:PLAY
     "o1l64s10m2000a"
1720 P=P+12:NEXT
1730 P=3:P1=P1+16:NEXT
1740 GOSUB 1430
1750 COLOR 1:PRESET(93,163):PRINT#1,"F3-
     Giro90"
1760 RETURN
1770 ' Espejo ▲
1780 COLOR 15:PRESET(93,173):PRINT#1,"F4
     -Espej▲"
1790 GOSUB 1520:GOSUB 2870
1800 P=3:P1=4
1810 FOR F=177 TO 170 STEP-1
1820 FOR N=45 TO 52
1830 IF POINT(N,F)=1 THEN PAINT(P,P1),3:
     LINE(P-1,P1-2)-(P+8,P1+11),4,B:PLAY
     "o5l32s10m300b"
1840 P=P+12:NEXT
1850 P=3:P1=P1+16:NEXT
1860 GOSUB 1430
1870 COLOR 1:PRESET(93,173):PRINT#1,"F4-
     Espej▲":RETURN
1880 ' Sprites selección
1890 PN=1
1900 COLOR 1:PRESET(35,200):PRINT#1,I
1910 LINE(91,141)-(165,200),5,BF
1920 LINE(X2,194)-(X2+13,207),15,B
1930 GOSUB 2760:PLAY"l10s12m300a":GOSUB
     1170
1940 E$=INKEY$
1950 A=STICK(0)
1960 LINE(X2,194)-(X2+13,207),15,B
1970 IF A=7 THEN GOSUB 2490
1980 IF A=3 THEN GOSUB 2660
1990 IF A=1 THEN GOSUB 2700
2000 IF A=5 THEN GOSUB 2730
2010 IF E$="½" THEN GOSUB 2390
2020 IF E$="¼" THEN GOSUB 3700
2030 IF E$="•" THEN RUN
2040 IF E$="n" THEN GOSUB 2820:GOSUB 292
     0:GOSUB 2820:RETURN
2050 IF E$=CHR$(11) THEN GOSUB 1520:GOSU
     B 2870:LINE(43,165)-(55,181),15,BF
2060 IF E$=CHR$(27) THEN GOSUB 2130
2070 IF E$="Z" THEN GOSUB 2530
2080 IF E$="r" THEN GOSUB 1230
2090 IF E$="f" THEN LINE(93,194)-(166,20
     7),5,BF:COLOR 1:GOSUB 2820:PLAY"l10
     s12m300b":GOSUB 1100:LINE(35,200)-(
     65,210),5,BF:PN=0:RETURN
2100 IF E$=CHR$(13)THEN GOSUB 2870:GOSUB
      2200:BEEP
2110 LINE(X2,194)-(X2+13,207),5,B
2120 GOTO 1940
2230 M=4:P=1:FOR F=DI+8*I TO 7+(DI+8*I)
2240 A$="00000000"+BIN$(VPEEK(F))
2250 A$=RIGHT$(A$,8):B$=""
2260 FOR N=1 TO 8:IF MID$(A$,N,1)="1" TH
     EN B$=B$+"█" ELSE B$=B$+" "
2270 NEXT
2280 BI=VAL("&B"+A$)
2290 H$=HEX$(BI):IF LEN(H$)=1 THEN H$="0
     "+H$
2300 COLOR 1:PRESET(120,Y1):PRINT#1,A$
2310 COLOR 1:PRESET(190,Y1):PRINT#1,H$:P
     RESET(208,Y1):PRINT#1,BI:PLAY"o8l16
     s8m300c"
2320 IF IM=1 THEN LPRINT B$;" ";A$;" ";H
     $;" ":BI
2330 A$="":B$="":Y1=Y1+10:Y2=Y2+1
```




```
2130 DI=30720:I1=I
2140 FOR NP=0 TO 255:I=NP:PRESET(35,200)
     :PRINT#1,NP;" "
2150 FOR M1=DI+8*NP TO 7+(DI+8*NP)
2160 IF VPEEK(M1)<>0 THEN GO=1
2170 NEXT
2180 IF GO=1 THEN GOSUB 2870:GOSUB 2200:
     GO=0
2190 NEXT:I=I1:PRESET(35,200):PRINT#1,I;
     " ":BEEP:GOSUB 2870:RETURN
2200 ' valores de sprite
2210 IF IM=1  THEN LPRINT"SPRITE    BINA
     RIO  H  DEC ":LPRINT
2220 A$="":B$="":Y1=20:Y2=0
2340 NEXT
2350 IF IM=1 THEN LPRINT"...............
     ..........."
2360 RETURN
2370 LINE(X2,194)-(X2+13,207),4,B:RETURN
2380 LINE(X2,194)-(X2+13,207),15,B:RETUR
     N
2390 ' almacen sprites
2400 COLOR 15:PRESET(93,143):PRINT#1,"F1
     -Guarda":GOSUB 2370
2410 A$="":VP=DC+8*I:PE=0:FOR N=4 TO 128
      STEP 16
2420 FOR M=3 TO 96 STEP 12
2430 IF POINT(M,N)=3 THEN A$=A$+"1"
```

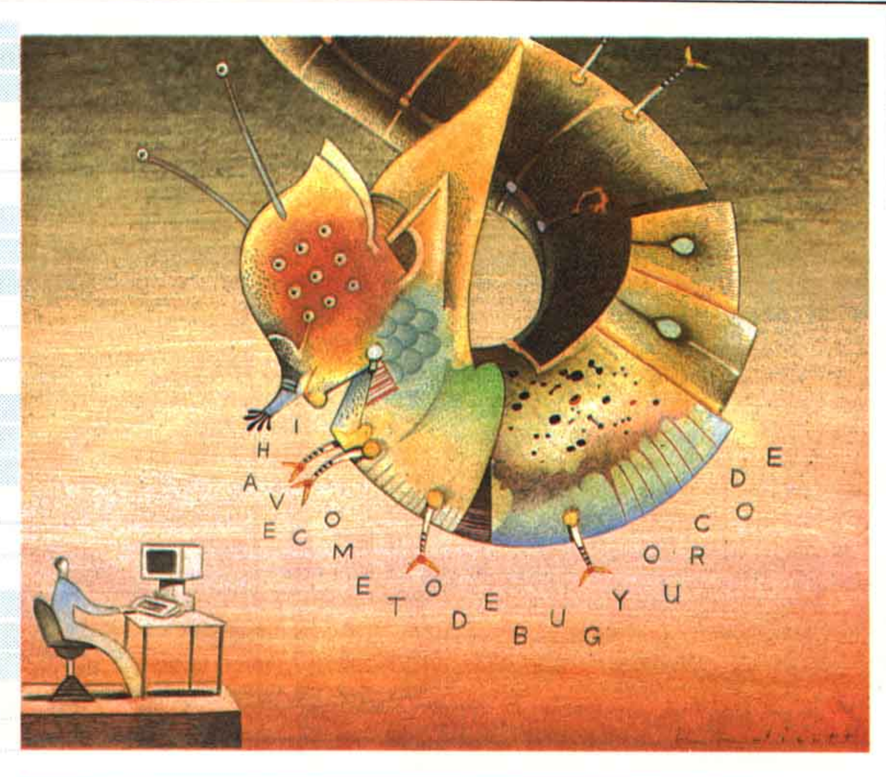
```
2440 IF POINT(M,N)=4 THEN A$=A$+"0"
2450 NEXT
2460 VPOKE VP+PE,VAL("&b"+A$):PLAY"l64a"
     ,"l64c","o8s12m300l64e"
2470 PE=PE+1:A$="":NEXT
2480 COLOR 1:PRESET(93,143):PRINT#1,"F1-
     Guarda":BEEP:GOSUB 2380:RETURN
2490 IF I<0 THEN I=0:RETURN
2500 IF X2<=93 THEN RETURN
2510 I=I-1:LINE(X2,194)-(X2+13,207),5,B:
     X2=X2-12:LINE(X2,194)-(X2+13,207),1
     5,B:PRESET(35,200):PRINT#1,I;" "
2520 PLAY"l64o3s1m100a":RETURN
2530 ' Copia sprite
2540 GOSUB 2370
2550 LINE(43,165)-(55,181),15,BF
```




```
2560 PRESET(93,173):COLOR 15:PRINT#1,"F4
     -Copia":GOSUB 1520
2570 DI=30720:M=4:P=1:FOR F=DI+8*I TO 7+
     (DI+8*I)
2580 D$="00000000"+BIN$(VPEEK(F))
2590 D$=RIGHT$(D$,8)
2600 FOR N=3 TO 96 STEP 12
2610 IF MID$(D$,P,1)="1" THEN PAINT (N,M
     ),3:LINE(N-1,M-2)-(N+8,M+11),4,B:PS
     ET(N/12+45,M/16+170),1:PLAY"l64o2s1
     2m300a"
2620 P=P+1:NEXT:M=M+16:P=1
2630 NEXT
2640 PRESET(93,173):COLOR 1:PRINT#1,"F4-
     Copia":GOSUB 2380:BEEP
2650 RETURN
2660 IF I>=255 THEN I=255:RETURN
2670 IF X2>=153 THEN RETURN
2680 I=I+1:LINE(X2,194)-(X2+13,207),5,B:
     X2=X2+12:LINE(X2,194)-(X2+13,207),1
     5,B:PRESET(35,200):PRINT#1,I;" "
2690 PLAY"l64o3s1m100a":RETURN
2700 IF I<=5 THEN RETURN
2710 I=I-6:P3=P3-6:PRESET(35,200):PRINT#
     1,I;" ":GOSUB 2760
2720 PLAY"l64o3s1m100a":RETURN
2730 IF I>=252 OR (X2=141 AND I=250) OR
     (X2=153 AND I=251) THEN RETURN
2740 I=I+6:P3=P3+6:PRESET(35,200):PRINT#
     1,I;" ":GOSUB 2760
2750 PLAY"l64o3s1m100a":RETURN
2760 P=96
2770 FOR N=0 TO 5
2780 IF P3+N>255 THEN RETURN
2790 PUT SPRITE N,(P,196),1,P3+N
2800 P=P+12:NEXT
2810 RETURN
2820 P=96
2830 S1=0:FOR N=B1 TO B1+31
2840 PUT SPRITE S1,(255,230),1,N
2850 P=P+12:S1=S1+1:NEXT:S1=0
2860 RETURN
2870 'Borrado pantalla num.
2880 LINE(116,11)-(184,109),5,BF
2890 LINE(187,11)-(205,109),5,BF
2900 LINE(208,11)-(248,109),5,BF
2910 RETURN
2920 ' Banco de pruebas
2930 X=100:Y=100:SP=1:CO=1:COLOR 1,12,12
     :CLS::DEF USR0=65:U=USR0(0):PLAY"o3
     l2s0m9000a"
2940 I1=0:SC=0:PO=0
2950 VDP(1)=SC+(VDP(1) AND 252)
2960 GOSUB 3550
2970 LINE(0,130)-(255,137),1,BF
2980 LINE(0,0)-(255,130),CC,BF
2990 LINE(0,0)-(255,130),15,B
3000 COLOR 1:PRESET(0,140):PRINT#1,"SPRI
     TE:"
3010 PRESET(74,140):PRINT#1,PR
3020 PRESET(0,153):PRINT#1,"F1-F2-Selec.
     sprite"
3030 PRESET(0,163):PRINT#1,"F3-Color Spr
     ite";CO(PR)
3040 PRESET(0,173):PRINT#1,"F4-Color Fon
     do ";CC
3050 PRESET(0,183):PRINT#1,"F5-Menú  BS-
     Animado"
3060 PRESET(0,193):PRINT#1,"SELECT-sprit
     es:":COLOR 15:PRESET(118,193):PRINT
     #1,B1;"a";B1+31
3070 COLOR 1:PRESET(0,203):PRINT#1,"ESC-
     Modo sprite:";Q$
3080 LINE(175,157)-(245,170),7,B
3090 GOSUB 3300
3100 DEF USR0=68:U=USR0(0)
3110 E$=INKEY$:A=STICK(0)
3120 IF E$="½" THEN GOSUB 3430
3130 IF E$="¼" THEN GOSUB 3470
3140 X%(PR)=X%(PR)-Q*(A=2 OR A=3 OR A=4)
     +Q*(A=6 OR A=7 OR A=8):IF X%(PR)>25
     5 THEN X%(PR)=-32
3150 IF X%(PR)<-32 THEN X%(PR)=255
3160 Y%(PR)=Y%(PR)-Q*(A=4 OR A=5 OR A=6)
     +Q*(A=8 OR A=1 OR A=2):IF Y%(PR)=>1
     22-PO THEN Y%(PR)=0
3170 IF Y%(PR)<0 THEN Y%(PR)=122-PO
3180 IF E$="%" THEN GOSUB 3390
3190 IF E$=CHR$(24) THEN GOSUB 3310
3200 IF E$=CHR$(27) THEN GOSUB 3600
3210 IF E$="n" THEN GOSUB 3520
3220 IF E$=CHR$(8) THEN GOSUB 5030
3230 IF E$="r" THEN PLAY"o3l2s0m9000a":D
     EF USR0=65:U=USR0(0):GOSUB 2820:GOS
     UB 540:VDP(1)=0+(VDP(1)AND252):RETU
     RN
3240 PUT SPRITE SF,(X%(PR),Y%(PR)),14,PR
3250 PUT SPRITE SF,(X%(PR),Y%(PR)),CO%(P
     R),PR
3260 IF E$<>"" THEN Q=Q+.5:GOTO 3110
```

```
3270 Q=1
3280 GET TIME PR$:COLOR 14:PRESET(180,16
     0):PRINT#1,PR$
3290 GOTO 3110
3300 S2=0:FOR F=B1 TO B1+31:PUT SPRITE S
     2,(X%(F),Y%(F)),CO%(F),F:S2=S2+1:NE
     XT:RETURN
3310 SF=0:PLAY"o3s14m900ad"
3320 B1=B1+32
3330 IF SC>1 AND B1>60 THEN B1=0
3340 IF B1>224 THEN B1=0
3350 GOSUB 3300:PRESET(118,193):COLOR 15
     :PRINT#1,B1;"a";B1+31;"        "
3360 PR=B1:PRESET(74,140):PRINT#1,PR;"
     "
3370 GOSUB 3530
3380 RETURN
3390 CC=CC+1:IF CC>15 THEN CC=1
3400 PLAY"o1s0m5000l64a":PRESET(120,173)
     :PRINT#1,CC
3410 LINE(1,1)-(254,129),CC,BF
3420 RETURN
3430 IF PR=<B1 THEN RETURN
3440 PR=PR-1:SF=SF-1:PRESET(74,140):PRIN
     T#1,PR
3450 GOSUB 3530
3460 RETURN
3470 IF SC>1 AND PR>63 THEN RETURN
3480 IF PR=>B1+31 THEN RETURN
3490 PR=PR+1:SF=SF+1:PRESET(74,140):PRIN
     T#1,PR
3500 GOSUB 3530
3510 RETURN
3520 CO%(PR)=CO%(PR)+1:PLAY"o8s1m900l64d
     ":IF CO%(PR)>15 THEN CO%(PR)=1
3530 PRESET(120,163):PRINT#1,CO%(PR)
3540 RETURN
3550 IF SC=0 THEN Q$="8x8   Normal  ":PO
     =0
3560 IF SC=1 THEN Q$="8x8   Ampliado":PO
     =10
3570 IF SC=2 THEN Q$="16x16 Normal  ":PO
     =10
3580 IF SC=3 THEN Q$="16x16 Ampliado":PO
     =22
3590 RETURN
3600 ' cambio de tipo
3610 DEF USR0=65:U=USR0(0):PLAY"l2o7s8m1
     970gae":GOSUB 2820
3620 SC=SC+1:IF SC>3 THEN SC=0
3630 IF B1>63 AND SC>1 THEN SC=0
3640 GOSUB 3550
3650 COLOR 1:PRESET(0,203):PRINT#1,"ESC-
     Modo sprite:";Q$
3660 VDP(1)=SC+(VDP(1) AND 252)
3670 GOSUB 2820:GOSUB 3300
3680 DEF USR0=68:U=USR0(0)
3690 RETURN
3700 ' Salvar cinta y disco
3710 DI=0:DF=0:GOSUB 2820
3720 COLOR 1,6,6:CLS:BEEP
3730 ON ERROR GOTO 4890
3740 DEF USR0=65:U=USR0(0)
3750 LINE(15,15)-(105,143),1,BF
3760 LINE(137,60)-(235,117),1,BF
3770 LINE(10,10)-(100,135),5,BF
3780 LINE(15,12)-(17,19),1,BF
3790 LINE(22,10)-(90,30),15,BF
3800 CIRCLE(55,70),17,14,,,1.34
3810 PAINT(55,70),14
3820 LINE(53,68)-(56,72),1,BF
3830 LINE(22,135)-(83,100),4,BF
3840 LINE(38,100)-(83,135),14,BF
3850 LINE(65,104)-(75,132),4,BF
3860 LINE(133,55)-(230,110),2,BF
3870 LINE(133,55)-(230,110),3,B
3880 LINE(148,21)-(218,31),1,BF
3890 LINE(145,18)-(215,28),14,BF
3900 COLOR,2:PRESET(138,60):PRINT#1,"F1-
     Salvar"
3910 PRESET(138,70):PRINT#1,"F2-Cargar"
3920 PRESET(138,80):PRINT#1,"F3-Listar"
3930 PRESET(138,90):PRINT#1,"F4-Borrar"
3940 PRESET(130,160):PRINT#1,"<- Sprite
     ->"
3950 PRESET(150,170):PRINT#1,"Space-Fija
     r"
3960 PRESET(138,100):PRINT#1,"F5-Menú"
3970 LINE(160,135)-(220,155),1,BF
3980 LINE(155,130)-(215,150),2,BF
3990 LINE(155,130)-(215,150),3,B
4000 I1=0
4010 PRESET(170,137):PRINT#1,I1
4020 DEF USR0=68:U=USR0(0)
4030 COLOR 1,15:N1=LEN(N$)-4:PRESET(25,1
     5):PRINT#1,LEFT$(N$,N1):COLOR1,2
4040 E$=INKEY$
4050 IF E$="½" THEN GOSUB 4150:GOTO 3700
4060 IF E$="¼" THEN GOSUB 4300:GOTO 3700
4070 IF E$="r" THEN GOSUB 4390:RETURN
4080 IF E$="n" THEN GOSUB 4590:GOTO 3700
4090 IF E$="L" THEN GOSUB 4780:GOTO 3700
4100 IF E$=CHR$(29) THEN COLOR,2:GOSUB 4
     410
4110 IF E$=CHR$(28) THEN COLOR,2:GOSUB 4
     460
4120 IF E$=" " THEN GOSUB 4510
4130 GET TIME PR$:COLOR 4,14:PRESET(150,
     20):PRINT#1,PR$
4140 GOTO 4040
4150 COLOR 7,1,1:CLS:SCREEN 0:SV=1
```

```
4160 PRINT SPC(9);"┌────────────────┐";SPC(20);"|SALVAR FICHEROS|";SPC(20);"└────────────────┘":PRINT:PRINT
4170 FILES"a:*.SPR":PRINT:PRINT
4180 PRINT"Salvar de sprite ";AS;" a ";BS;"(S/N)":PRINT
4190 E$=INKEY$
4200 IF E$="s" THEN GOTO 4230
4210 IF E$="n" THEN SCREEN 5:RETURN
4220 GOTO 4190
4230 GOSUB 4530
4240 GOSUB 4550
4250 N$=N$+".SPR"
4260 ON ERROR GOTO 4890
4270 SCREEN 5
4280 BSAVE N$,DI,DF,S
4290 RETURN
4300 COLOR 7,1,1:CLS:SCREEN 0
4310 PRINT SPC(9);"┌────────────────┐";SPC(20);"|CARGAR FICHEROS|";SPC(20);"└────────────────┘":PRINT:PRINT
4320 FILES"a:*.SPR":PRINT:PRINT
4330 GOSUB 4530
4340 GOSUB 4550
4350 N$=N$+".SPR"
4360 SCREEN 5
4370 BLOAD N$,S
4380 RETURN
4390 DEF USR0=65:U=USR0(0)
4400 GOSUB 540:RETURN
4410 COLOR 11:PRESET(130,160):PRINT#1,"<"
4420 I1=I1-1:IF I1<=0 THEN I1=0
4430 COLOR 1:PRESET(170,137):PRINT#1,I1
4440 COLOR 1:PRESET(130,160):PRINT#1,"<"
4450 IF INKEY$<>"" THEN GOTO 4410 ELSE RETURN
4460 COLOR 11:PRESET(218,160):PRINT#1,">"
4470 I1=I1+1:IF I1>=255 THEN I1=255
4480 COLOR 1:PRESET(170,137):PRINT#1,I1
4490 COLOR 1:PRESET(218,160):PRINT#1,">"
4500 IF INKEY$<>"" THEN GOTO 4460 ELSE RETURN
4510 IF FJ=0 THEN DI=I1:AS=I1:FJ=1:PRESET(0,190):PRINT#1,"Desde sprite:";DI;" ":DI=DC+8*DI:BEEP:RETURN
4520 IF FJ=1 THEN DF=I1:BS=I1:FJ=0:PRESET(0,200):PRINT#1,"Hasta sprite:";DF;" ":DF=(DC+8*DF)+7:BEEP:RETURN
4530 IF DF<DI THEN SWAP DF,DI
4540 RETURN
4550 COLOR 7,1
4560 INPUT"NOMBRE>";N$
4570 IF LEN(N$)>8 THEN GOTO 4560
4580 RETURN
4590 COLOR 7,1,1:SCREEN 0
4600 PRINT "┌──────────────────┐ ┌─────────────┐ |LISTAR FICHEROS| |I--IMPRESORA| └─────────────┘ |P--PANTALLA |";SPC(23);"└─────────────┘"
4610 E$=INKEY$
4620 IF E$="i" THEN GOTO 4720
4630 IF E$="p" THEN GOTO 4650
4640 GOTO 4610
4650 CLS:PRINT"Espere un momento":PRINT
4660 FILES"a:*.SPR"
4670 LOCATE 0,0:PRINT"                "
4680 FOR N=2 TO 21:LOCATE 12,N:PRINT"≠":NEXT
4690 LOCATE 0,22:PRINT"<Pulse una tecla>"
4700 IF INKEY$="" THEN GOTO 4700
4710 SCREEN 5:RETURN
4720 LOCATE 0,3:PRINT"Conecte la impresora"
4730 LPRINT"FICHEROS DE SPRITES":LPRINT
4740 LFILES"a:*.SPR"
4750 LOCATE 0,22:PRINT"<<Pulse una tecla>>"
4760 IF INKEY$="" THEN GOTO 4760
4770 SCREEN 5:RETURN
4780 COLOR 7,1,1:SCREEN 0
4790 PRINT SPC(9);"┌────────────────┐";SPC(20);"|BORRAR FICHEROS|";SPC(20);"└────────────────┘":PRINT:PRINT
4800 FILES"a:*.SPR"
4810 PRINT:PRINT:INPUT"¿CUAL BORRO";N$
4820 N$=N$+".spr":KILL N$
4830 IF N$<>SPACE$(12) THEN PRINT:PRINT N$;" borrado.":PRINT
4840 PRINT"¿Quieres borrar otro?"
4850 E$=INKEY$
4860 IF E$="s" THEN N$="":GOTO 4780
4870 IF E$="n" THEN N$="":SCREEN 5:RETURN
4880 GOTO 4850
4890 'rutina errores
4900 IF SV=1 AND ERR<>68 THEN SV=0:RESUME NEXT
4910 SCREEN 0:COLOR 15,6,6
4920 IF ERR=66 THEN ER$="Disco lleno"OK
4930 IF ERR=690 THEN ER$="Error I/O"
4940 IF ERR=70 THEN ER$="No hay disco"
4950 IF ERR=68 THEN ER$="Disco protegido"
4960 IF ERR=53 THEN ER$="No existe fiche
```



```
    ro "+N$
4970 IF ERR=56 THEN ER$="Nombre incorrec
    to"
4980 PLAY"s10m1000abc","s10m1000cde","o3
    fga"
4990 CX=(37-LEN(ER$))/2
5000 LOCATE CX,10:PRINT ER$:N$=SPACE$(12
    ):FOR F=1TO1000:NEXT
5010 SCREEN 5:RESUME 5020
5020 RETURN
5030 ' animación
5040 PLAY"s1m1000o6a","s10m1000c"
5050 COLOR 1,CC
5060 IF CC=1 THEN COLOR 15
5070 EN=0
5080 LINE(167,146)-(248,186),1,BF
5090 LINE(160,141)-(244,181),CC,BF
5100 LINE(159,140)-(245,182),15,B
5110 PRESET(161,142):PRINT#1,"Desde:":PR
    ESET(205,142):PRINT#1,N1:PRESET(161
    ,152):PRINT#1,"-----":PRESET(165,16
    2):PRINT#1,"<CURSORES>":PRESET(180,
    172):PRINT#1,"Y SPACE"
5120 E$=INKEY$
5130 IF E$=CHR$(29) THEN GOSUB 5340
5140 IF E$=CHR$(28) THEN GOSUB 5380
5150 IF E$=CHR$(32) THEN GOSUB 5420
5160 IF EN=2 THEN COLOR 1,12:RETURN
5170 GOTO 5120
5180 IF N1>N2 THEN SWAP N1,N2
5190 PRESET(163,145):PRINT#1,"▲":PRESET(
    163,155):PRINT#1,"▼":PRESET(163,170
    ):PRINT#1,"BS"
5200 E$=INKEY$
5210 FOR F=N1 TO N2
5220 PUT SPRITE 0,(194,145),CO%(F),F
5230 FOR M=1 TO PA:NEXT
5240 IF E$=CHR$(31) THEN PA=PA+1
5250 IF E$=CHR$(30) THEN PA=PA-1
5260 IF E$=CHR$(8) THEN GOTO5290
5270 NEXT
5280 GOTO 5200
5290 PUT SPRITE 0,(X%(PR),Y%(PR)),CO%(PR
    ),0
5300 LINE(159,140)-(248,186),12,BF
5310 LINE(175,157)-(245,170),7,B
5320 PLAY"s1m100o6a","s10m100c"
5330 RETURN
5340 IF EN=0 THEN N1=N1-1:IF N1=<0 THEN
    N1=0
5350 IF EN=1 THEN N2=N2-1:IF N2=<0 THEN
    N2=0
5360 PRESET(205,142):PRINT#1,N1:PRESET(2
    05,152):PRINT#1,N2
5370 RETURN
5380 IF EN=0 THEN N1=N1+1:IF N1=>255 THE
    N N1=255
5390 IF EN=1 THEN N2=N2+1:IF N2=>255 THE
    N N2=255
5400 PRESET(205,142):PRINT#1,N1:PRESET(2
    05,152):PRINT#1,N2
5410 RETURN
5420 IF EN=1 THEN LINE(160,141)-(244,181
    ),CC,BF:GOSUB 5180:EN=2:RETURN
5430 EN=1:BEEP
5440 PRESET(161,152):PRINT#1,"Hasta:":PR
    ESET(205,152):PRINT#1,N2
5450 PLAY"s1m1000o6a","s10m1000c"
5460 RETURN
```

CURSO DE INGLES

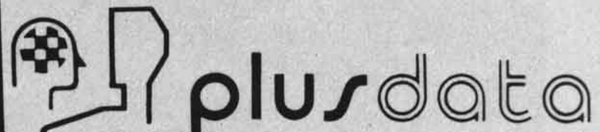
The Gruneberg Linkword Language System es un sistema, para enseñanza de idiomas, más rápido y fácil que los métodos convencionales aplicados actualmente.
En poco tiempo, máximo 20 horas, te enseñará un vocabulario de 400 palabras y adquirirás unas buenas nociones de gramática. Esto te permitirá entender y ser entendido en tus viajes a lugares de habla inglesa o en tus contactos con personas que se expresen en ese idioma.
Por otra parte, el Sistema PlusData, consigue que el ordenador se convierta en un perfecto profesor que te explicará, orientará y corregirá, manteniendo en todo momento un "diálogo" interactivo de resultados sorprendentes.



–L. Taylor. **"POPULAR COMPUTER WORLD":**
"Quedé francamente atónito al comprobar la efectividad de la sugestión de imágenes como elemento de ayuda a la retención..."

–**"PERSONAL COMPUTER WORLD":**
"Un suceso fuera de serie..."

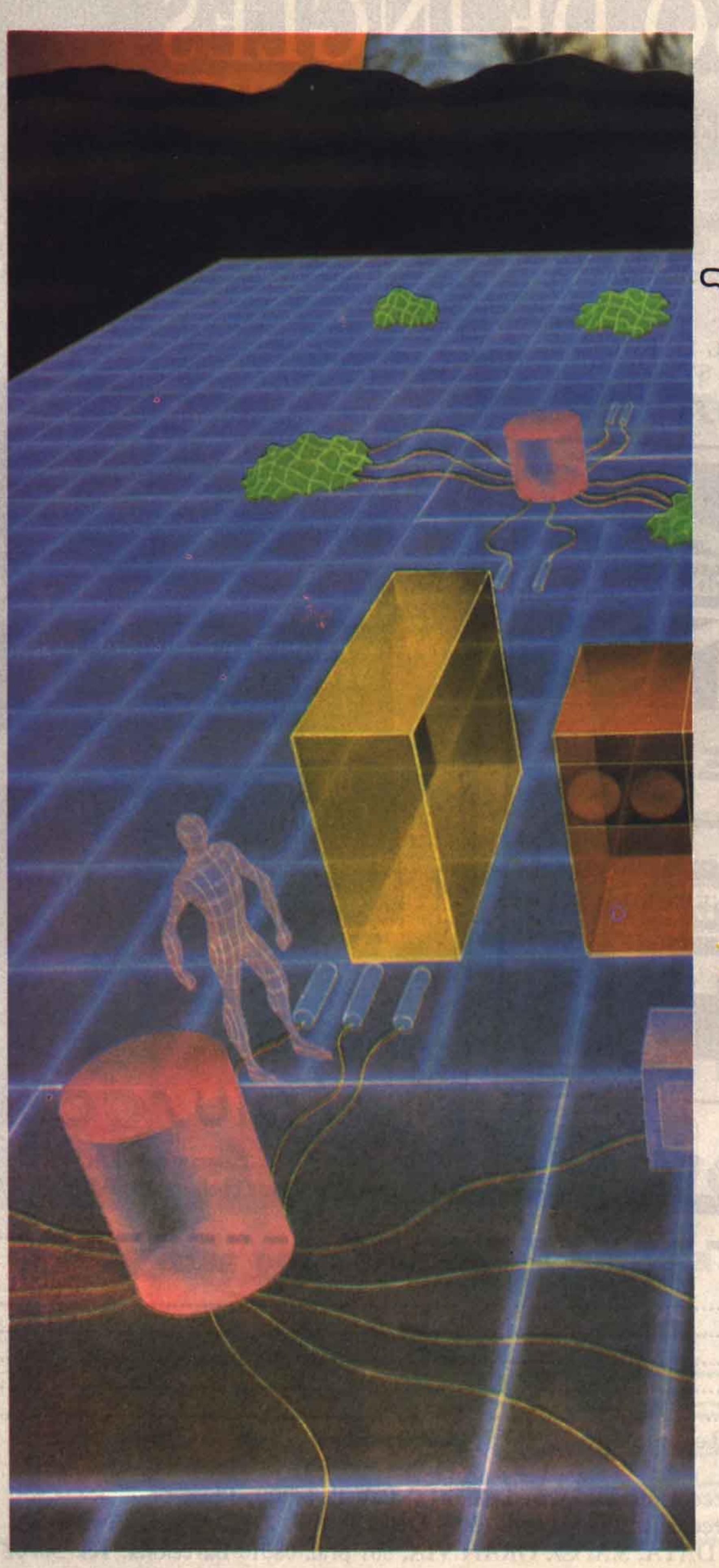
–Bill Barnet. **"COMPUTER CHOICE":**
"De todos los paquetes para aprender idiomas éste es el más interesante..."



Programas de EAO para EGB.
Cursos de Basic, Cobol, etc. AUTODIDACTAS.

Nombre ....
Apellidos ....
Dirección ....
Población ....
D.P. .... Tlno. ....
Forma de pago: Reembolso ☐ Giro postal ☐ Envío talón ☐

☐ Curso de Inglés 1.ª parte. 10 lecciones Linkword. (Cinta) P.V.P. 6.900.-Ptas.
☐ Curso de Inglés 1.ª parte. 10 lecciones Linkword. (3,5"-Disk) P.V.P. 7.900.-Ptas.
ENVIAR ESTE CUPON A: PLUS DATA, S.A. C/. GRAN VIA, 661 pral. 08010-Barcelona. Tel. 246 02 02



BASI

Un programa en *BASIC* sirve para realizar repetidamente una serie de sentencias encaminadas a obtener unos resultados concretos. Pero para que un programa *BASIC* se ejecute es necesario introducir una serie de datos. Podemos hacer esto de varias formas y, vamos ahora a ver cómo hacerlo desde el teclado.

En la mayoría de los programas se precisa introducir ciertos datos desde el teclado. En BASIC MSX existen varias instrucciones para hacer esto. La primera de ellas es la instrucción *INPUT*, que tiene la siguiente forma:

INPUT ["MENSAJE";]VARIABLE 1 [,VARIABLE 2,....]

donde MENSAJE es precisamente eso, un mensaje que podemos incluir en la sentencia *INPUT* y que aparecerá en el monitor. Generalmente, se usa para recordarnos cuáles son las variables que se deben introducir. MENSAJE es opcional (incluiremos entre corchetes todo lo que sea opcional) y debe ir entre comillas. VARIABLE 1 es cualquier variable aceptada por BASIC MSX (entera, real o alfanumérica) y es obligatoria: cada vez que se utiliza una sentencia INPUT es preceptivo el uso de al menos una variable; pero con esta sentencia se pueden utilizar tantas variables como permita una línea de programa (recuerda que una línea de programa puede tener un máximo de 255 caracteres). Por tanto, la VARIABLE 1 es obligatoria, pero el resto de las variables son opcionales. Veamos un ejemplo de utilización de esta sentencia: Supongamos que en nuestro programa vamos a introducir desde el teclado el nombre de una persona y su número de teléfono; para ello utilizamos una sentencia



Introducir datos desde el teclado y desde un fichero de acceso secuencial

INPUT de la siguiente forma:

10 INPUT "Nombre, Teléfono";N$,T%

que, una vez ejecutada hará que aparezca en el monitor lo siguiente:

Nombre, Teléfono ?

a lo que debemos contestar con el nombre, una coma y el número de teléfono, así:

José Pérez, 1234567

la coma le indica al intérprete *BASIC* que lo que está escrito antes de ella corresponde a una variable y lo que está escrito a continuación corresponde a otra.

La sentencia *INPUT* permite introducir datos tanto numéricos como alfanuméricos separados por comas, por tanto la coma no puede incluirse como parte de un dato alfanumérico. Si, por ejemplo, escribimos:

10 INPUT "Dirección",D$

no podremos contestar:

Calle del Pez, 15

porque el intérprete entenderá que Calle del Pez es un dato y 15 otro y responderá con el mensaje de error.

Extra ingnored

que quiere decir que, como sólo estaba esperando un dato para asignarlo a la variable D$, ignora el segundo que se le ha dado (el 15).

No está permitido escribir, por ejemplo:

10 INPUT "Nombre";N$, "Teléfono";T%

ya que el intérprete nos dará un *Syntax error in 10* como respuesta. Esto se debe a que no se pueden poner varios mensajes en una sentencia *INPUT*. Por tanto, todos los mensajes deberán ir al principio de la sentencia.

Sin embargo, es posible asignar la coma y otros caracteres especiales a una variable alfanumérica con ayuda de la sentencia:

LINE INPUT ["MENSAJE";] VARIABLE ALFANUMERICA

Con esta sentencia sólo se puede introducir una variable. Si quieres introducir varias variables deberás utilizar la sentencia *LINE INPUT* tantas veces como variablesj tengas. Un ejemplo de utilización de *LINE INPUT* es el siguiente:

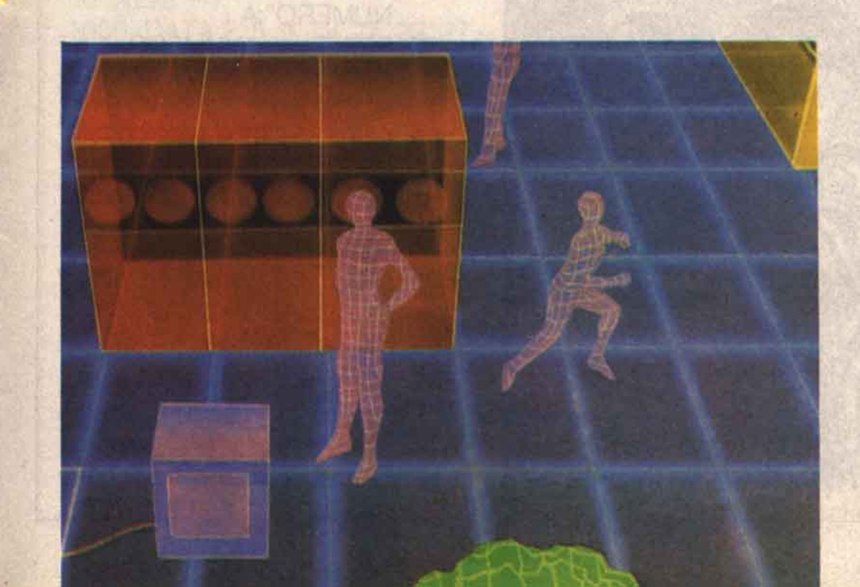
10 LINE INPUT "Dirección";D$

a lo que ahora sí podrás responder:

Calle del Pez, 15

La razón de esto es que cuando el intérprete *BASIC* se encuentra con esta sentencia, asigna a la variable absolutamente todos los caracteres que se intoduzcan hasta que se pulsa la tecla *(RETURN)* (retorno de carro). Como pudiste ver en el artículo sobre ficheros de acceso secuencial que publicamos en MSX de Noviembre esta setencia es útil para leer una línea de texto procedente de un fichero en cinta o en disco. La sentencia:

LINE INPUT #1,A$



recoge del fichero #1 todos los caracteres ASCII que encuentre hasta que lea un código de retorno de carro (código 13).

Veamos ahora para qué sirve la sentencia *INPUT$*, que tiene la forma:

VARIABLE ALFANUMERICA= INPUT$(N)

Aquí el intérprete *BASIC* esperad hasta que se pulsen tantos caracteres como indica el número N (que también puede ser una variable), y los asigna a la variable alfanumérica situada a la izquierda del signo igual. Un ejemplo de utilización de *INPUT$* es el siguiente:

```
10 INPUT "¿HAS
   TERMINADO... S/N?"
20 A$=INPUT$(1)
30 IF A$="S" OR A$="s"
   THE END
40 GOTO 10
```

aquí el intérprete escribe el mensaje de la línea 10 y espera a que pulses una tecla. Si la tecla pulsada es la S (mayúscula o minúscula), el programa terminará con la sentencia *END* de la línea a 30 y si pulsas cualquier otra tecla volverá a aparecer el mensaje de la línea 10. *INPUT$* te permite además asignar caracteres especiales a una variable. El siguiente programa te da como salida la tecla que has pulsado y el código ASCII de la misma de forma que si pulsas la tecla A mayúsculaf te da como salida

A 65

y si pulsas la tecla *(RETURN)*, te responde con una línea en blanco y el número 13. Prueba tú mismo:

```
10 A$=INPUT$(1)
20 PRINT A$;ASC(A$)
30 GOTO 10
```

Otra forma de introducir un carácter en el ordenador es haciendo uso de la sentencia *INKEY$*, que tiene la siguiente forma:

VARIABLE ALFANUMERICA= INKEY$

Cuando el intérprete pasa por una sentencia *INKEY$*, no espera a que pulses alguna tecla, simplemente comprueba si hay una tecla pulsada y si es así, asigna el código ASCII correspondiente a la misma a la variable situada a la izquierda del signo igual. Por eso, cuando aparece una sentencia *INKEY$* suele ir acompañada de otra condicional, commo en el siguiente ejemplo:

```
10 A$=INKEY$
20 IF A$=" " THEN 10
```

De esta forma, si no se ha pulsado ninguna tecla (es decir si en A$ no hay nada) se vuelve a ejecutar la sentencia *INKEY$* hasta que se haya pulsado alguna.

Haciendo uso de esta sentencia, podemos también everiguar los códigos ASCII de todas las teclas:

```
10 A$=INKEY$
20 IF A$=" " THEN 10
30 PRINT A$;ASC(A$)
40 GOTO 10
```

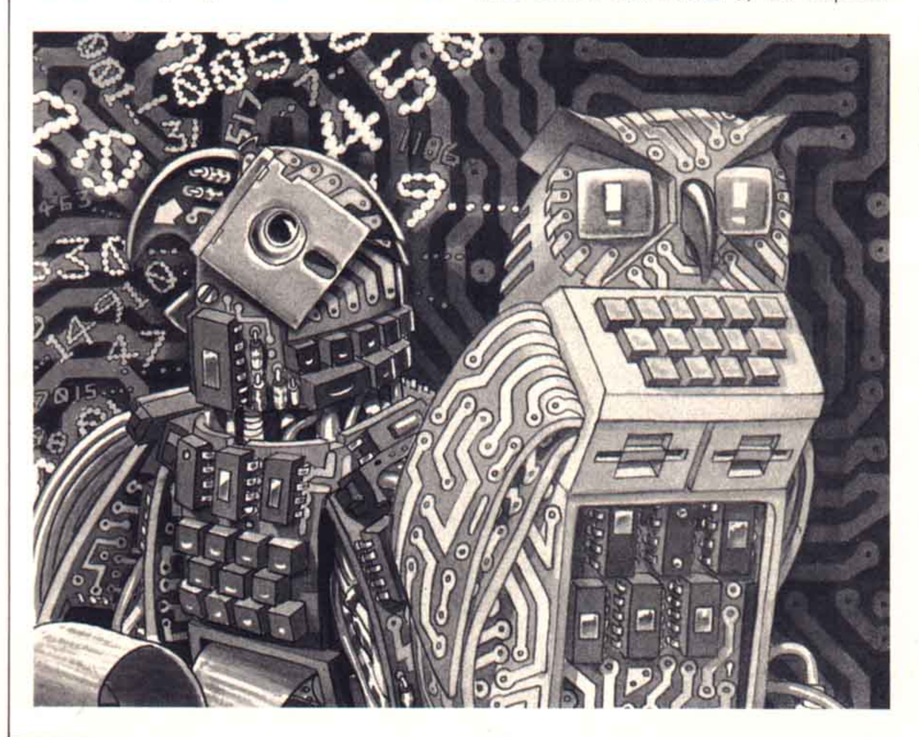
Por último, existe una forma de incluir los datos a utilizar por un programa dentro del mismo programa, y es mediante el uso de líneas *DATA*. Supongamos que queremos hallar el cuadrado y la raíz cuadrada de 10 números. Podemos utilizar el siguiente programa:

```
10 INPUT "INTRODUCE UN
   NÚMERO";A
20 PRINT "NUMERO=";A;"
   CUADRADO=";A.A; RAIZ=";
   SQR(A)
30 GOTO 10
```

o bien este otro:

```
10 RESTORE 1000
20 READ A
30 PRINT "NUMERO=";A;"
   CUADRADO=";A.A;"
   RAIZ="; SQR(A)
40 GOTO 20
1000 DATA 2, 5, 9, 12, 21, 23,
     27, 34, 36, 49
```

la sentencia *RESTORE* de la línea



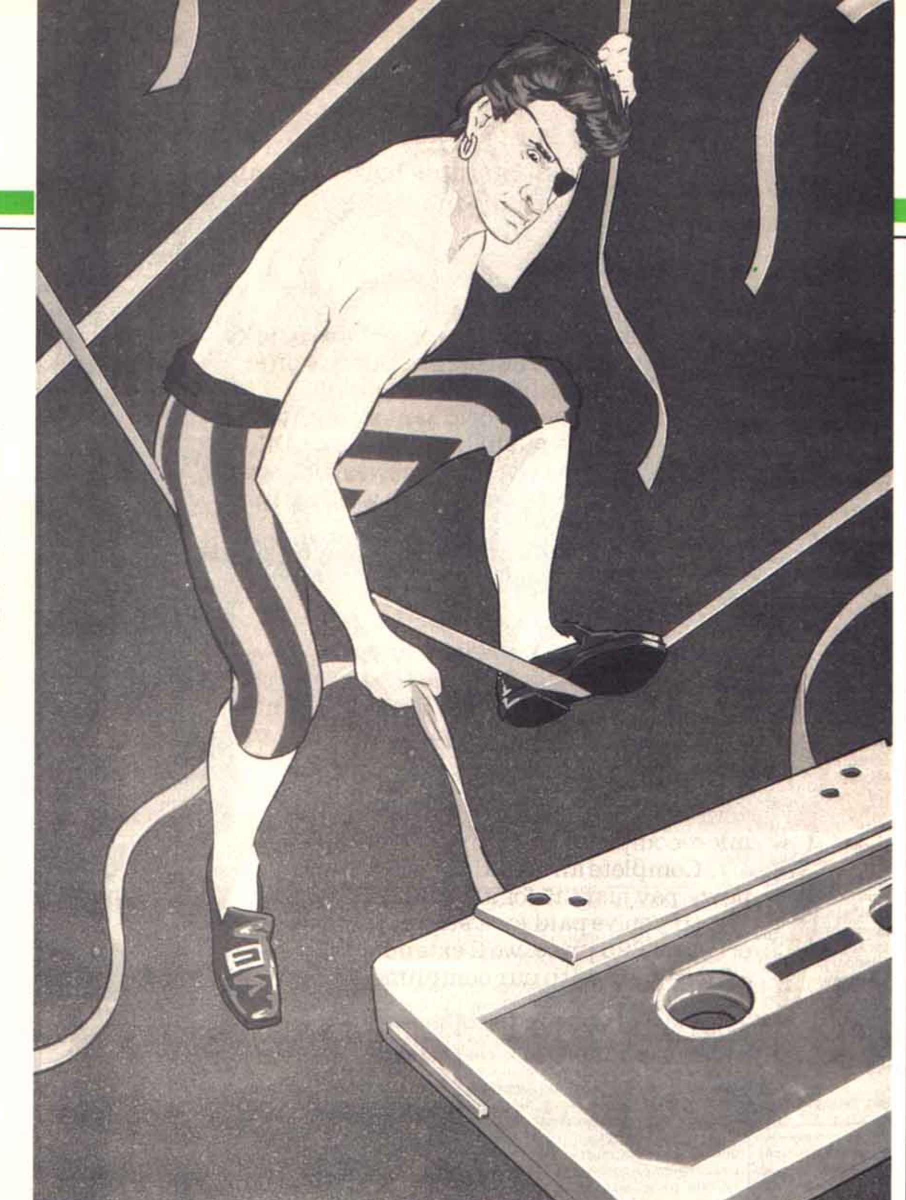
10 quiere decir que los próximos datos que se van a leer son los que están a partir de la línea 1000 (*restore* quiere decir restablecer). Entonces, cada vez que el programa pasa por la línea 20 se lee un nuevo dato; la primera vez se leerá el primer dato, la segunda el segundo... y así sucesivamente hasta que se termina la lista. La línea *DATA* indica al intérprete que todo lo que viene a continuación en esa línea son datos. Por tanto, este programa va leyendo los datos de la línea 1000 e imprime el dato, su cuadrado y su raíz cuadrada, para a continuación leer un nuevo dato. Al acabarse los datos, el programa termina con el siguiente mensaje de error:

```
OUT OF DATA IN 20
```

que quiere decir que al pasar por la sentencia *READ* de la línea 20, no se encuentran más datos. Esto se puede arreglar con un pequeño bucle, ya que siempre que incluimos datos en una línea *DATA* sabemos cuántos son:

```
10 RESTORE 1000
20 FOR I%=1 TO 10
30 READ A
40 PRINT      "NUMERO=";A;"
   CUADRADO=";A*A=RAIZ="'
   SQR(A)
50 NEXT I%
1000 DATA 2, 5, 9, 12, 21, 23, 27,
   34, 36, 49
```

Bien pues con esto está casi todo dicho sobre cómo introducir datos desde el teclado. Sólo nos resta hablar algo de cómo introducir datos procedentes de un fichero secuencial. En realidad la forma en que se hace esto es bastante parecida a cómo se hace desde el teclado. Se usan las mismas sentencias, aunque indicando, en cada caso, el número de fichero del que se lee. Así, la sentencia *INPUT* para leer de un fichero de acceso



secuencial es de la forma:

```
INPUT # n,VARIABLE 1
[VARIABLE 2,...]
```

donde n indica el nombre del fichero abierto en la sentencia *OPEN FOR INPUT.* La diferencia con la sentencia *INPUT* para leer del teclado es que ahora no es preciso incluir ningún mensaje ya que al fichero en cinta no le hace falta que el recordemos qué es lo que va a leer.

De forma parecida, la sntencia *LINE INPUT* cuando se vaa a leer de un fichero de acceso secuencial tiene la forma:

```
LINE INPUT # n,VARIABLE
ALFANUMERICA
```

y *BASIC* tomará caracteres del fichero hasta que se encuentre un retorno de carro.

La sentencia *INPUT$* para leer de un fichero es ahora:

```
A$=INPUT$(X, # n)
```

donde X indica el número de caracteres a leer y n es el número del fichero.

La sentencia *INKEY$* no se utiliza con ficheros, ni, lógicamente, las sentencias *READ.. DATA.*

Así hemos visto todas las formas de introducir datos en el ordenador tanto a partir del teclado como de un fichero de acceso secuencial. En otro número de MSX Magazine veremos todo lo relativo a ficheros de acceso directo en disco, incluyendo cómo leer datos y cómo escribirlos en el mismo.

J. Antonio Feberero

A l igual que los MSX, los SVI-318 y SVI-328 usan como generador de sonido o PSG el AY-3-8910.

Seguramente es ya una de las facilidades que más conocen y usan tanto los que tienen un SVI-318 ó un SVI-328 como los que manejan un MSX ya que, aparte de ser una de las características mejor explicadas en el manual, se publicó en los números 7 y 8 de esta revista un completo artículo sobre el mismo como componente de un ordenador MSX.

Para ver el funcionamiento del PSG a través de sus registros sin repetir aquí otra vez lo mismo, podéis introducir en vuestro ordenador el programa que viene a continuación, basado en el comando SOUND.

La única diferencia en lo que al PSG se refiere entre los ordenadores con norma MSX y los SVI-318 y 328 radica en los *ports* que se utilizan para su conexión con el Z-80.

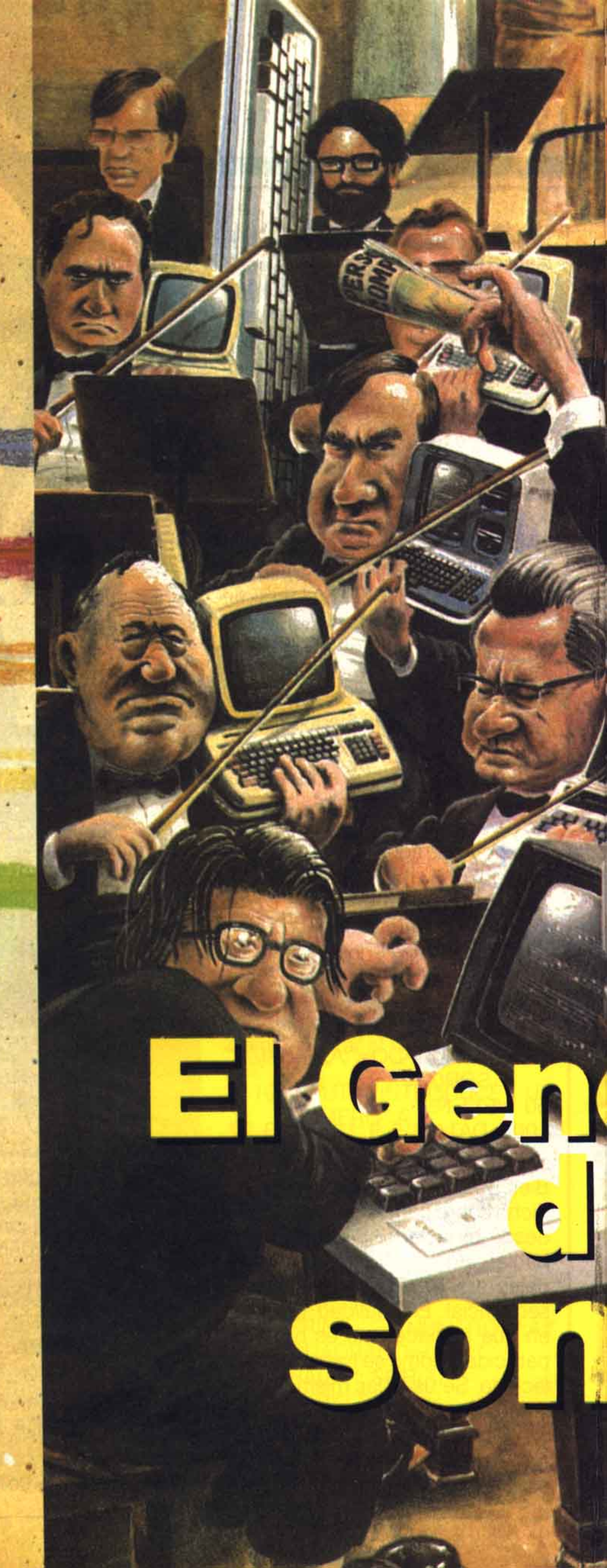
Como podemos ver en la figura 1 del capítulo anterior los *ports* utilizados son tres:

— El *port* 88 hex. es un *port* de salida en el que colocaremos el número del registro del PSG al que queremos acceder.

— El *port* 8C hex., también de salida será el que envíe el dato a colocar en el registro señalado por el port anterior.

— El *port* 90 hex. de entrada es el que recoge el contenido del registro al que señala el *port* 88 hex.

Los procedimientos para leer y escribir en los registros del PSG



El Gene d son



desde el BASIC, aparte del ya conocido comando *SOUND* son pues como sigue:

```
10 REM Lectura del PSG.
20 OUT&h88,registro a acceder.
30 contenido del registro =
   INP(&h90)
10 REM Escritura del PSG
20 OUT&h88,regisitro a acceder
30 OUT&h8c,valor a escribir
```

Pero estos *ports* nos van a permitir hacer más cosas que música. Si habéis ejecutado el programa 1 habréis observado que no aparecen los registros 14 y 15 del *chip* de sonido, esto se debe a que dichos registros son *ports* de entrada y salida del PSG y no están relaccionados con la creación de sonidos.

En el primer artículo dedicado al SV-328 en el número 12 de la revista ya usamos el registro 15 para conmutar bancos de memoria. Vamos a ver en el siguiente apartado cómo se usa el registro 14 ó *port A* del PSG para leer el joystick.

A través del *port* A del generador de sonidos podemos leer la información que nos llega de los *joysticks* o palancas de mando conectadas a los *buses* laterales numerados como 1 y 2. Las teclas de movimiento del cursor, aunque pueden ser utilizadas como otro *joystick*, son consideradas incluidas en el teclado y se manejan como tal a través de los *ports* del PPI, de igual forma que los disparadores de las palancas.

Como en los otros registros del PSG la información que nos inte-

resa será leida mediante la colocación en el *port* 88 hex del número de registro a acceder para luego recoger en el *port* 90 hex la información requerida.

Tampoco debemos olvidar que en el registro 8 del PSG los *bits* 6 y 7 del mismo son los que indican respectivamente que los *ports* A y B están dispuestos para entrada (*bit*=0) o para salida (*bit*=1). Así la secuencia completa para leer la información de los *joysticks* queda como sigue:

```
10 OUT&h88,8:A=INP(&h90):
   SOUND8,&B01111111 AND A
20 OUT&h88,14
30 X=INP(&H90)
```

Con lo que obtenemos en la variable «X» un número que interpretaremos *bit* a *bit* como sigue:

— Si todos los *bits* son 1 indica que los mandos 1 y 2 están en posición de reposo, o sea, que no está accionado ninguno de los *joysticks*.

— *Bit* 0 = 0. El *joystick* 1 está accionado hacia adelante.

— *Bit* 1 = 0. El *joystick* 1 está accionado hacia atrás.

— *Bit* 2 = 0. El *joystick* 1 está accionado a la izquierda.

— *Bit* 3 = 0. El *joystick* 1 está accionado hacia la derecha.

— *Bit* 4 = 0. El *joystick* 2 está accionado hacia adelante.

— *Bit* 5 = 0. El *joystick* 2 está accionado hacia atrás.

— *Bit* 6 = 0. El *joystick* 2 está accionado a la izquierda.

— *Bit* 7 = 0. El *joystick* 2 está accionado hacia la derecha.

Hay que tener en cuenta que para un mismo mando pueden estar accionadas dos posiciones a la vez, con lo que el resultado será un accionamiento en diagonal.

Para hacer una lectura directa de los disparadores del *stick* usaremos el *port* &h98 que es el *port* A del PPI según la figura 1, así si usamos la instrucción

```
10 X=INP (&h98)
```

tendremos en X un número cuyos *bits* 4 y 5 indicarán el estado de los disparadores según el siguiente concepto:

— *Bit* 4 = 0. Ha sido pulsado el disparador del *stick* 1.

— *Bit* 5 = 0. Ha sido pulsado el disparador del *stick* 2.

Lectura directa del teclado

El BASIC de nuestro ordenador tiene una instrucción que detecta la pulsación de una tecla, esta instrucción es

INKEY$

la cual, no obstante no detecta la pulsación de todas las teclas existentes como podrás comprobar si introduces las siguientes líneas de programa.

```
10 A$=INKEY$:IFA$=" "THEN10
20 PRINT ASC(A$):GOTO10
```

Tras pulsar *'RUN'* y *'ENTER'* observarás que la pulsación de casi todas las teclas produce la aparición de un número en la pantalla, el cual es el código ASCII de dicha tecla. Pero hay algunas teclas que no producen efecto alguno *(CTRL, SHIFT, LEFT-GRPH, SELECT,* etc.), y no obstante dichas teclas también

```
10 '******SUPERSINTE**********
20 DIMA%(13):LOCATE,,0:GOSUB150
30 '******MOVIMIENTO DEL CURSOR*******
40 D=STICK(0):IFD=0THEN90
50 IFD=1THENLOCATE33,E+2:PRINT"ｯ":E=E-1:IFE<0THENE=0
60 IFD=5THENLOCATE33,E+2:PRINT"ｯ":E=E+1:IFE>13THENE=13
70 LOCATE33,E+2:PRINT"ノ":FORA=1TO50:NEXTA
80 '******ACTIVACION DEL BIT PULSADO********
90 N$=INKEY$:IFN$=""THEN40
100 IFASC(N$)<48ORASC(N$)>55THEN40
110 N=VAL(N$):A%(E)=A%(E)XOR(2^N):LOCATE35,E+2:PRINTUSING"###";A%(E)
120 A$=BIN$(A%(E)):A$=STRING$(8-LEN(A$),48)+A$:LOCATE16+2*(7-N),E+2
130 PRINTMID$(A$,8-N,1):SOUNDE,A%(E):GOTO40
140 '*******DIBUJO DE PANTALLA*************
150 CLS:PRINTSPC(16);"7 6 5 4 3 2 1 0":PRINT:FORC=0TO13:READC$
160 LOCATE0,C+2:PRINTUSING" ##:";C;:PRINTC$
170 LOCATE16,C+2:PRINT"0 0 0 0 0 0 0 0  ｯ   0":NEXT
180 PRINT:PRINT:PRINT"  CURSOR VERTICAL PARA ELEGIR REGISTRO"
190 PRINT"  0 A 7 PARA ACTIVAR O DESACTIVAR BIT":RETURN
200 DATATONO A,,TONO B,,TONO C,,RUIDO,MEZCLADOR,VOLUMEN A,VOLUMEN B,VOLUMEN C,EN
VOL.DUR.,,ENVOL.FORM
```

PROGRAMA 1 Utilización gráfica de los registros del PSG

son detectadas por el ordenador.

El medio por el que nosotros podemos acceder a ellas es mediante una lectura directa del teclado a través de los *ports* del PPI 96 hex y 99 hex, y guiándonos según la tabla de la figura 2.

Notas a la figura 2:

I. La Y es el número de fila y el valor Z es igual a 2 elevado al número de columna.

II. Las dos filas inferiores se refieren al teclado numérico.

III. Las casillas *CUP, LFT, DWN* y *RGT* se refieren a las teclas de dirección del cursor.

IV. Las teclas pulsadas con *SHIFT, CTRL,* etc. producen el mismo resultado que si se pulsaran solas por lo que cuando queramos un efecto conjunto de varias teclas a la vez debemos leer todas y cada una de las teclas implicadas.

El método a seguir es el siguiente:

— Enviamos a través del *port* 96 hex el número Y de la matriz de la figura 2 que corresponde a la fila en que está la tecla a detectar pero OReado con 16.

— En el *port* 99 hex leemos un valor cuyos bits tienen una correspondencia directa con una tecla según la matriz de la figura 2. Cada cero que aparezca en la representación binaria de este valor será una tecla pulsada.

— Ejemplo: Queremos detectar las teclas pulsadas en la fila 8 de la figura 2. Las líneas a incluir en nuestro programa serán como sigue:

```
10 OUT&h96,80R16
20 X=INP (&H90)
```

Para ver el número en binario hacemos un:

```
X$=aBIN$(X):X$=STRING$
(8—LEN(X$),48)+X$:PRINTX$
```

con lo que veremos incluso los ceros no significativos del valor.

Si el valor resultante es por ejemplo

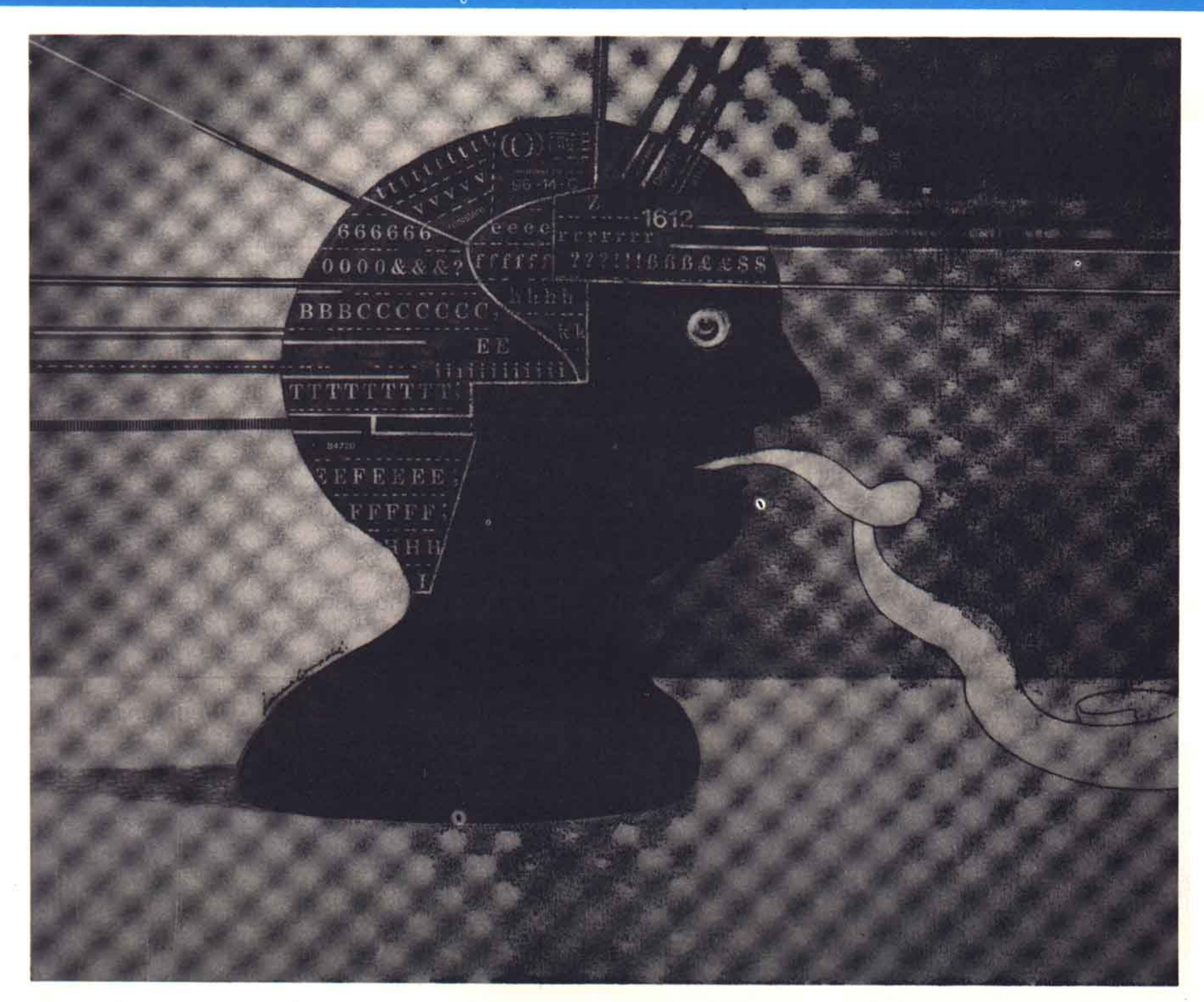
11010111 bin

nos indicará que está encendida la tecla *CAPS-LOCK* y estamos pulsando la tecla *PRINT.*

Venerando Solís

MATRIZ PARA EL TECLADO DEL SVI
==============================

Y\\Z	128	64	32	16	8	4	2	1
0	7	6	5	4	3	2	1	0
1	/	.	=	,	'	:	9	8
2	g	f	e	d	c	b	a	-
3	o	n	m	l	k	j	i	h
4	w	v	u	t	s	r	q	p
5	CUP	BS	]	\	[	z	y	x
6	LFT	ENT	STP	ESC	RG	LG	CTR	SFT
7	DWN	INS	CLS	F5	F4	F3	F2	F1
8	RGT		PRT	SEL	CAP	DEL	TAB	SPC
9	7	6	5	4	3	2	1	0
10	,	.	/	*	-	+	9	8



Desensamblador

El presente programa es un desensamblador, ocupa unas 19 K, y puede desensamblar cualquier instrucción del Z-80.

Su función más inmediata es la de desensamblar las rutinas del sistema, aunque también se puede hacer lo mismo con cualquier programa en código máquina que coloquemos por encima de la posición &HD000.

A la hora de utilizarlo hay que tener en cuenta dos cosas:

a) Desensamblar únicamente áreas de memoria que contengan programas en código máquina, no las áreas de datos que estos usen.

b) Los números que van de &H8000 a &HFFFF se corresponden con —32768 a —1 en decimal.

Cuando el programa se ejecuta, pide como datos la primera y última dirección de la zona de memoria a desensamblar, y a partir de ese momento irán apareciendo en pantalla dos columnas, la de la derecha contendrá los nemónicos y la de la izquierda la dirección en la que se encuentra la instrucción.

Para detener o reanudar la ejecución, utilizar la tecla de STOP.

Pasemos ahora a ver la estructura del programa:

Las líneas 80-120 tienen como misión llamar a la rutina que deba tratar cada una de las instrucciones que se vayan encontrando en la memoria. Estas rutinas son:

NOT (CB, ED, DD, FD)

Trata las instrucciones que comienzan por un número distinto de &HCB, &HED, &HDD o &HFD.

En las líneas 160-200, se calcula cuál es el «DATA» correspondiente. A continuación se escriben la dirección y el nemónico, y si es necesario se escribe también el dato o dirección al que haga referencia la instrucción, de lo cual se encargan las líneas 230-720. Por ejemplo: JP E007.

CB

Las instrucciones que empiezan por &HCB no hacen referencia a datos numéricos ni direcciones, por tanto sólo es preciso identificar el «DATA» que se corresponde con el contenido de la posición de memoria y escribir la dirección y el nemónico.

ED

Esta rutina es muy parecida a la primera, sólo se diferencia en que las instrucciones que se tratan aquí comienzan por &HED. Su estructura es la siguiente:

Líneas 1190-1320: encontrar el nemónico.

Líneas 1350-1420: calcular dato o dirección que le acompañan.

DD o FD

Las instrucciones que empiezan por &HDD o &HFD son las que manejan los registros de índice «IX» e «IY».

Esta rutina, que es la más larga (líneas 1520-2950) se encarga de escribir directamente el nemónico que corresponda y los datos numéricos a los que afecte.

Cada una de las 4 rutinas anteriores es independiente de las otras 3 y puede ser suprimida siempre que no vaya a utilizarse.

Existen dos rutinas más, una (líneas 2960-3000) se encarga de escribir los nemónicos, y la otra (3010-3050) escribe los datos o direcciones en formato hexadecimal.

Joaquínn F. Hernández

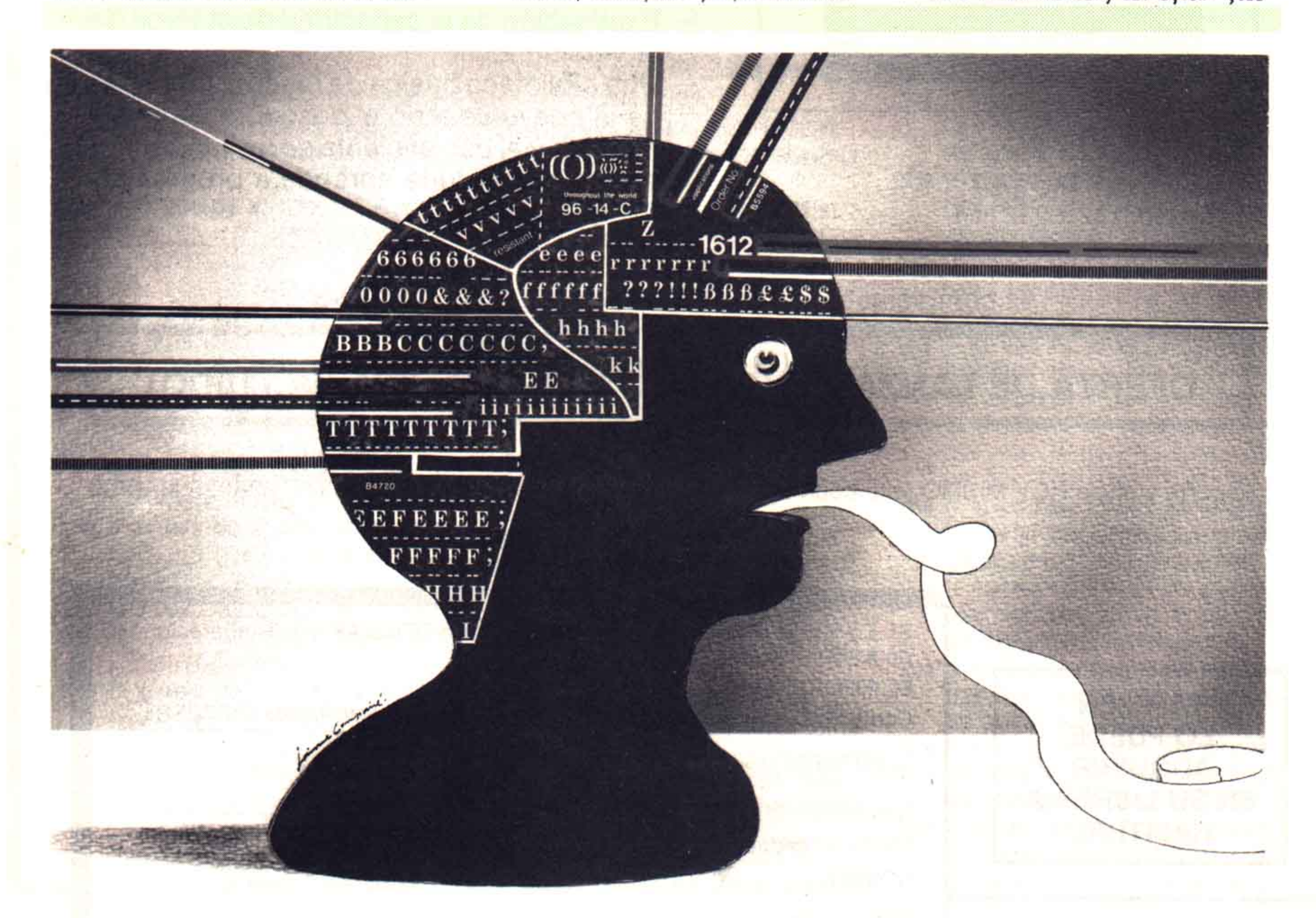
```
10 REM DESENSAMBLADOR
20 CLS:CLEAR 200,&HD000
30 LOCATE 11,1:PRINT"DESENSAMBLADOR"
40 LOCATE 11,2:PRINT"=============="
50 LOCATE 1,4:PRINT"(Todos los datos num
     èricos en hex.)"
60 LOCATE 5,8:INPUT"Direcciòn de comienz
     o";DI$:DI$="&H"+DI$:DI=VAL(DI$)
70 LOCATE 5,10:INPUT"Ultima direcciòn
      ";DF$:DF$="&H"+DF$:DF=VAL(DF$)
80 IF PEEK(DI)<>&HCB AND PEEK(DI)<>&HED
     AND PEEK(DI)<>&HDD AND PEEK(DI)<>&H
     FD THEN GOSUB 160
90 IF PEEK(DI)=&HCB THEN GOSUB 950
100 IF PEEK(DI)=&HED THEN GOSUB 1190
110 IF PEEK(DI)=&HDD OR PEEK(DI)=&HFD TH
     EN GOSUB 1520
120 IF DI>DF THEN END ELSE GOTO 80
130 '********************************
140 'NOT(CB,ED,DD,FD)
150 '********************************
160 I=PEEK(DI)
170 IF PEEK(DI)>&HCB THEN I=I-1
180 IF PEEK(DI)>&HDD THEN I=I-1
190 IF PEEK(DI)>&HED THEN I=I-1
200 IF PEEK(DI)>&HFD THEN I=I-1
210 PRINT TAB(5);HEX$(DI);
220 RESTORE 760:GOSUB 2960:L=1:SUM=0
230 IF PEEK(DI)=&H1 THEN GOSUB 3010:PRIN
     T:SUM=3:L=0
240 IF PEEK(DI)=&H6 THEN GOSUB 3030:PRIN
     T:SUM=2:L=0
250 IF PEEK(DI)=&HE THEN GOSUB 3030:PRIN
     T:SUM=2:L=0
260 IF PEEK(DI)=&H10 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
270 IF PEEK(DI)=&H11 THEN GOSUB 3010:PRI
     NT:SUM=2:L=0
280 IF PEEK(DI)=&H16 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
290 IF PEEK(DI)=&H18 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
300 IF PEEK(DI)=&H1E THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
310 IF PEEK(DI)=&H20 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
320 IF PEEK(DI)=&H21 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
330 IF PEEK(DI)=&H22 THEN GOSUB 3010:PRI
     NT"),HL":SUM=3:L=0
340 IF PEEK(DI)=&H26 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
350 IF PEEK(DI)=&H28 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
360 IF PEEK(DI)=&H2A THEN GOSUB 3010:PRI
     NT")":SUM=3:L=0
370 IF PEEK(DI)=&H2E THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
380 IF PEEK(DI)=&H30 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
390 IF PEEK(DI)=&H31 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
400 IF PEEK(DI)=&H32 THEN GOSUB 3010:PRI
     NT"),A":SUM=3:L=0
410 IF PEEK(DI)=&H36 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
420 IF PEEK(DI)=&H38 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
430 IF PEEK(DI)=&H3A THEN GOSUB 3010:PRI
     NT")":SUM=3:L=0
440 IF PEEK(DI)=&H3E THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
450 IF PEEK(DI)=&HC2 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
460 IF PEEK(DI)=&HC3 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
470 IF PEEK(DI)=&HC4 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
480 IF PEEK(DI)=&HC6 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
490 IF PEEK(DI)=&HCA THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
500 IF PEEK(DI)=&HCC THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
510 IF PEEK(DI)=&HCD THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
520 IF PEEK(DI)=&HCE THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
530 IF PEEK(DI)=&HD2 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
540 IF PEEK(DI)=&HD3 THEN GOSUB 3030:PRI
     NT"),A":SUM=2:L=0
550 IF PEEK(DI)=&HD4 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
560 IF PEEK(DI)=&HD6 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
570 IF PEEK(DI)=&HDA THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
580 IF PEEK(DI)=&HDB THEN GOSUB 3030:PRI
     NT")":SUM=2:L=0
590 IF PEEK(DI)=&HDC THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
600 IF PEEK(DI)=&HDE THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
610 IF PEEK(DI)=&HE2 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
620 IF PEEK(DI)=&HE4 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
630 IF PEEK(DI)=&HE6 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
```

```
640 IF PEEK(DI)=&HEA THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
650 IF PEEK(DI)=&HEC THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
660 IF PEEK(DI)=&HEE THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
670 IF PEEK(DI)=&HF2 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
680 IF PEEK(DI)=&HF4 THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
690 IF PEEK(DI)=&HF6 THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
700 IF PEEK(DI)=&HFA THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
710 IF PEEK(DI)=&HFC THEN GOSUB 3010:PRI
     NT:SUM=3:L=0
720 IF PEEK(DI)=&HFE THEN GOSUB 3030:PRI
     NT:SUM=2:L=0
730 IF L=1 THEN PRINT:DI=DI+1 ELSE DI=DI
     +SUM
740 RETURN
750 '*****DATA NOT(CB,DD,ED,FD)*****
760 DATA "NOP","LD BC,","LD(BC),A","INC
     BC","INC B","DEC B","LD B,","RLCA",
     "EX AF,AF'","ADD HL,BC","LD A,(BC)"
     ,"DEC BC","INC C","DEC C","LD C,","
     RRCA"
770 DATA "DJNZ ","LD DE,","LD (DE),A","I
     NC DE","INC D","DEC D","LD D,","RLA
     ","JR ","ADD HL,DE","LD A,(DE)","DE
     C DE","INC E","DEC E","LD E,","RRA"
780 DATA "JR NZ,","LD HL,","LD (","INC H
     L","INC H","DEC H","LD H,","DAA","J
     R Z,","ADD HL,HL","LD HL,(","DEC HL
     ","INC L","DEC L","LD L,","CPL"
790 DATA "JR NC,","LD SP,","LD (","INC S
     P","INC (HL)","DEC (HL)","LD (HL),"
     ,"SCF","JR C,","ADD HL,SP","LD A,("
     ,"DEC SP","INC A","DEC A","LD A,","
     CCF"
800 DATA "LD B,B","LD B,C","LD B,D","LD
     B,E","LD B,H","LD B,L","LD B,(HL)",
     "LD B,A","LD C,B","LD C,C","LD C,D"
     ,"LD C,E","LD C,H","LD C,L","LD C,(
     HL)","LD C,A"
810 DATA "LD D,B","LD D,C","LD D,D","LD
     D,E","LD D,H","LD D,L","LD D,(HL)",
     "LD D,A","LD E,B","LD E,C","LD E,D"
     ,"LD E,E","LD E,H","LD E,L","LD E,(
     HL)","LD E,A"
820 DATA "LD H,B","LD H,C","LD H,D","LD
     H,E","LD H,H","LD H,L","LD H,(H,L)"
     ,"LD H,A","LD L,B","LD L,C","LD L,D
     ","LD L,E","LD L,H","LD L,L","LD L,
     (HL)","LD L,A"
830 DATA "LD (HL),B","LD (HL),C","LD (HL
     ),D","LD (HL),E","LD (LD),H","LD (H
     L),L","HALT","LD (HL),A","LD A,B","
     LD A,C","LD A,D","LD A,E","LD A,H",
     "LD A,L","LD A,(HL)","LD A,A"
840 DATA "ADD A,B","ADD A,C","ADD A,D","
     ADD A,E","ADD A,H","ADD A,L","ADD A
     ,(HL)","ADD A,A","ADC A,B","ADC A,C
     ","ADC A,D","ADC A,E","ADC A,H","AD
     C A,L","ADC A,(HL)","ADC A,A"
850 DATA "SUB B","SUB C","SUB D","SUB E"
     ,"SUB H","SUB L","SUB (HL)","SUB A"
     ,"SBC A,B","SBC A,C","SBC A,D","SBC
      A,E","SBC A,H","SBC A,L","SBC A,(H
     L)","SBC A,A"
860 DATA "AND B","AND C","AND D","AND E"
     ,"AND H","AND L","AND (HL)","AND A"
     ,"XOR B","XOR C","XOR D","XOR E","X
     OR H","XOR L","XOR (HL)","XOR A"
870 DATA "OR B","OR C","OR D","OR E","OR
      H","OR L","OR (HL)","OR A","CP B",
     "CP C","CP D","CP E","CP H","CP L",
     "CP (HL)","CP A"
880 DATA "RET NZ","POP BC","JP NZ,","JP
     ","CALL NZ,","PUSH BC","ADD A,","RS
     T 00","RET Z","RET","JP Z,","CALL Z
     ,","CALL ","ADC A,","RST 08"
890 DATA "RET NC","POP DE","JP NC,","OUT
     (","CALL NC,","PUSH DE","SUB ","RST
      10","RET C","EXX","JP C,","IN A,("
     ,"CALL C,","SBC A,","RST 18"
900 DATA "RET PO","POP HL","JP PO,","EX
     (SP),HL","CALL PO,","PUSH HL","AND
     ","RST 20","RET PE","JP (HL)","JP P
     E,","EX DE,HL","CALL PE,","XOR ","R
     ST 28"
910 DATA "RET P","POP AF","JP P,","DI","
     CALL P,","PUSH AF","OR ","RST 30","
     RET M","LD SP,HL","JP M,","EI","CAL
     L M,","CP ","RST 38"
920 '******************************
930 ' CB
940 '******************************
950 I=PEEK(DI+1)
960 IF PEEK(DI+1)>&H2F THEN I=I-8
970 PRINT TAB(5);HEX$(DI);
980 RESTORE 1000:GOSUB 2960:PRINT:DI=DI+
     2:RETURN
990 '*****DATA CB*****
1000 DATA "RLC B","RLC C","RLC D","RLC E
     ","RLC H","RLC L","RLC (HL)","RLC A
     ","RRC B","RRC C","RRC D","RRC E","
     RRC H","RRC L","RRC (HL)","RRC A"
1010 DATA "RL B","RL C","RL D","RL E","R
     L H","RL L","RL (HL)","RL A","RR B"
     ,"RR C","RR D","RR E","RR H","RR L"
     , RR (HL)","RR A"
1020 DATA "SLA B","SLA C","SLA D","SLA E
     ","SLA H","SLA L","SLA (HL)","SLA A
     ","SRA B","SRA C","SRA D","SRA E","
     SRA H","SRA L","SRA (HL)","SRA A"
1030 DATA "SRL B","SRL C","SRL D","SRL E
     ","SRL H","SRL L","SRL (HL)","SRL A
     "
1040 DATA "BIT 0,B","BIT 0,C","BIT 0,D",
     "BIT 0,E","BIT 0,H","BIT 0,L","BIT
     0,(HL)","BIT 0,A","BIT 1,B","BIT 1,
     C","BIT 1,D","BIT 1,E","BIT 1,H","B
     IT 1,L","BIT 1,(HL)","BIT 1,A"
1050 DATA "BIT 2,B","BIT 2,C","BIT 2,D",
     "BIT 2,E","BIT 2,H","BIT 2,L","BIT
     2,(HL)","BIT 2,A","BIT 3,B","BIT 3,
     C","BIT 3,D","BIT 3,E","BIT 3,H","B
     IT 3,L","BIT 3,(HL)","BIT 3,A"
1060 DATA "BIT 4,B","BIT 4,C","BIT 4,D",
     "BIT 4,E","BIT 4,H","BIT 4,L","BIT
     4,(HL)","BIT 4,A","BIT 5,B","BIT 5,
     C","BIT 5,D","BIT 5,E","BIT 5,H","B
     IT 5,L","BIT 5,(HL)","BIT 5,A"
1070 DATA "BIT 6,B","BIT 6,C","BIT 6,D",
     "BIT 6,E","BIT 6,H","BIT 6,L","BIT
     6,(HL)","BIT 6,A","BIT 7,B","BIT 7,
     C","BIT 7,D","BIT 7,E","BIT 7,H","B
     IT 7,L","BIT 7,(HL)","BIT 7,A"
1080 DATA "RES 0,B","RES 0,C","RES 0,D",
     "RES 0,E","RES 0,H","RES 0,L","RES
     0,(HL)","RES 0,A","RES 1,B","RES 1,
     C","RES 1,D","RES 1,E","RES 1,H","R
     ES 1,L","RES 1,(HL)","RES 1,A"
1090 DATA "RES 2,B","RES 2,C","RES 2,D",
     "RES 2,E","RES 2,H","RES 2,L","RES
     2,(HL)","RES 2,A","RES 3,B","RES 3,
     C","RES 3,D","RES 3,E","RES 3,H","R
     ES 3,L","RES 3,(HL)","RES 3,A"
1100 DATA "RES 4,B","RES 4,C","RES 4,D",
     "RES 4,E","RES 4,H","RES 4,L","RES
     4,(HL)","RES 4,A","RES 5,B","RES 5,
     C","RES 5,D","RES 5,E","RES 5,H","R
     ES 5,L","RES 5,(HL)","RES 5,A"
1110 DATA "RES 6,B","RES 6,C","RES 6,D",
     "RES 6,E","RES 6,H","RES 6,L","RES
     6,(HL)","RES 6,A","RES 7,B","RES 7,
     C","RES 7,D","RES 7,E","RES 7,H","R
     ES 7,L","RES 7,(HL)","RES 7,A"
1120 DATA "SET 0,B","SET 0,C","SET 0,D",
     "SET 0,E","SET 0,H","SET 0,L","SET
     0,(HL)","SET 0,A","SET 1,B","SET 1,
     C","SET 1,D","SET 1,E","SET 1,H","S
     ET 1,L","SET 1,(HL)","SET 1,A"
1130 DATA "SET 2,B","SET 2,C","SET 2,D",
     "SET 2,E","SET 2,H","SET 2,L","SET
     2,(HL)","SET 2,A","SET 3,B","SET 3,
     C","SET 3,D","SET 3,E","SET 3,H","S
     ET 3,L","SET 3,(HL)","SET 3,A"
1140 DATA "SET 4,B","SET 4,C","SET 4,D",
     "SET 4,E","SET 4,H","SET 4,L","SET
     4,(HL)","SET 4,A","SET 5,B","SET 5,
     C","SET 5,D","SET 5,E","SET 5,H","S
     ET 5,L","SET 5,(HL)","SET 5,A"
```

```
1150 DATA "SET 6,B","SET 6,C","SET 6,D",
     "SET 6,E","SET 6,H","SET 6,L","SET
     6,(HL)","SET 6,A","SET 7,B","SET 7,
     C","SET 7,D","SET 7,E","SET 7,H","S
     ET 7,L","SET 7,(HL)","SET 7,A"
1160 '*****************************
1170 'ED
1180 '*****************************
1190 I=PEEK(DI+1)
1200 IF PEEK(DI+1)>&H3F THEN I=I-64
1210 IF PEEK(DI+1)>&H4C THEN I=I-1
1220 IF PEEK(DI+1)>&H4E THEN I=I-1
1230 IF PEEK(DI+1)>&H55 THEN I=I-2
1240 IF PEEK(DI+1)>&H5D THEN I=I-2
1250 IF PEEK(DI+1)>&H66 THEN I=I-3
1260 IF PEEK(DI+1)>&H6E THEN I=I-3
1270 IF PEEK(DI+1)>&H71 THEN I=I-1
1280 IF PEEK(DI+1)>&H77 THEN I=I-4
1290 IF PEEK(DI+1)>&H9F THEN I=I-36
1300 IF PEEK(DI+1)>&HA7 THEN I=I-4
1310 IF PEEK(DI+1)>&HAF THEN I=I-4
1320 IF PEEK(DI+1)>&HB7 THEN I=I-4
1330 PRINT TAB(5);HEX$(DI);:DI=DI+1
1340 RESTORE 1460:GOSUB 2960:L=1:SUM=0
1350 IF PEEK(DI)=&H43 THEN GOSUB 3010:PR
     INT"),BC":SUM=3:L=0
1360 IF PEEK(DI)=&H4B THEN GOSUB 3010:PR
     INT")":SUM=3:L=0
1370 IF PEEK(DI)=&H53 THEN GOSUB 3010:PR
     INT"),DE":SUM=3:L=0
1380 IF PEEK(DI)=&H5B THEN GOSUB 3010:PR
     INT")":SUM=3:L=0
1390 IF PEEK(DI)=&H63 THEN GOSUB 3010:PR
     INT"),HL":SUM=3:L=0
1400 IF PEEK(DI)=&H6B THEN GOSUB 3010:PR
     INT")":SUM=3:L=0
1410 IF PEEK(DI)=&H73 THEN GOSUB 3010:PR
     INT"),SP":SUM=3:L=0
1420 IF PEEK(DI)=&H7B THEN GOSUB 3010:PR
     INT")":SUM=3:L=0
1430 IF L=1 THEN PRINT:DI=DI+1 ELSE DI=D
     I+SUM
1440 RETURN
1450 '*****DATA ED*****
1460 DATA "IN B,(C)","OUT (C),B","SBC HL
     ,BC","LD (","NEG","RETN","IM 0","LD
      I,A","IN C,(C)","OUT (C),C","ADC H
     L,BC","LD BC,(","RETI","LD R,A","IN
      D,(C)","OUT (C),D","SBC HL,DE","LD
      (","IM 1","LD A,I","IN E,(C)","OUT
      (C),E","ADC HL,DE","LD DE,(","IM 2
     "
1470 DATA "LD A,R","IN H,(C)","OUT (C),H
     ","SBC HL,HL","LD (","RRD","IN L,(C
     )","OUT (C),L","ADC HL,HL","LD HL,(
     ","RLD","IN F,(C)","SBC HL,SP","LD
     (","IN A,(C)","OUT (C),A","ADC HL,S
     P","LD SP,("
1480 DATA "LDI","CPI","INI","OUTI","LDD"
     ,"CPD","IND","OUTD","LDIR","CPIR","
     INIR","OTIR","LDDR","CPDR","INDR","
     OTDR"
1490 '*****************************
1500 'DD OR FD
1510 '*****************************
1520 N1=PEEK(DI):N2=PEEK(DI+1):N3=PEEK(D
     I+2):N4=PEEK(DI+3)
1530 PRINT TAB(5);HEX$(DI);:DI=DI+1
1540 IF N1=&HDD AND N2=&H8E THEN PRINT T
     AB(12);"ADC A,(IX+";:GOSUB 3030:PRI
     NT")":DI=DI+2:RETURN
1550 IF N1=&HFD AND N2=&H8E THEN PRINT T
     AB(12);"ADC A,(IY+";:GOSUB 3030:PRI
     NT")":DI=DI+2:RETURN
1560 IF N1=&HDD AND N2=&H86 THEN PRINT T
     AB(12);"ADD A,(IX+";:GOSUB 3030:PRI
     NT")":DI=DI+2:RETURN
1570 IF N1=&HFD AND N2=&H86 THEN PRINT T
     AB(12);"ADD A,(IY+";:GOSUB 3030:PRI
     NT")":DI=DI+2:RETURN
1580 IF N1=&HDD AND N2=&HA6 THEN PRINT T
     AB(12);"AND (IX+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
1590 IF N1=&HFD AND N2=&HA6 THEN PRINT T
     AB(12);"AND (IY+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
1600 IF N1=&HDD AND N2=&HCB AND N4=&H46
     THEN PRINT TAB(12);"BIT 0,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1610 IF N1=&HFD AND N2=&HCB AND N4=&H46
     THEN PRINT TAB(12);"BIT 0,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1620 IF N1=&HDD AND N2=&HCB AND N4=&H4E
     THEN PRINT TAB(12);"BIT 1,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1630 IF N1=&HFD AND N2=&HCB AND N4=&H4E
     THEN PRINT TAB(12);"BIT 1,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1640 IF N1=&HDD AND N2=&HCB AND N4=&H56
     THEN PRINT TAB(12);"BIT 2,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1650 IF N1=&HFD AND N2=&HCB AND N4=&H56
     THEN PRINT TAB(12);"BIT 2,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1660 IF N1=&HDD AND N2=&HCB AND N4=&H5E
     THEN PRINT TAB(12);"BIT 3,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1670 IF N1=&HFD AND N2=&HCB AND N4=&H5E
     THEN PRINT TAB(12);"BIT 3,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1680 IF N1=&HDD AND N2=&HCB AND N4=&H66
     THEN PRINT TAB(12);"BIT 4,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1690 IF N1=&HFD AND N2=&HCB AND N4=&H66
     THEN PRINT TAB(12);"BIT 4,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1700 IF N1=&HDD AND N2=&HCB AND N4=&H6E
     THEN PRINT TAB(12);"BIT 5,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1710 IF N1=&HFD AND N2=&HCB AND N4=&H6E
     THEN PRINT TAB(12);"BIT 5,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1720 IF N1=&HDD AND N2=&HCB AND N4=&H76
     THEN PRINT TAB(12);"BIT 6,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1730 IF N1=&HFD AND N2=&HCB AND N4=&H76
     THEN PRINT TAB(12);"BIT 6,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1740 IF N1=&HDD AND N2=&HCB AND N4=&H7E
     THEN PRINT TAB(12);"BIT 7,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1750 IF N1=&HFD AND N2=&HCB AND N4=&H7E
     THEN PRINT TAB(12);"BIT 7,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
1760 IF N1=&HDD AND N2=&HBE THEN PRINT T
     AB(12);"CP (IX+";:GOSUB 3030:PRINT"
     )":DI=DI+2:RETURN
1770 IF N1=&HFD AND N2=&HBE THEN PRINT T
     AB(12);"CP (IY+";:GOSUB 3030:PRINT"
     )":DI=DI+2:RETURN
1780 IF N1=&HDD AND N2=&H35 THEN PRINT T
     AB(12);"DEC (IX+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
1790 IF N1=&HFD AND N2=&H35 THEN PRINT T
     AB(12);"DEC (IY+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
1800 IF N1=&HDD AND N2=&H9 THEN PRINT TA
     B(12);"ADD IX,BC":DI=DI+1:RETURN
1810 IF N1=&HDD AND N2=&H19 THEN PRINT T
     AB(12);"ADD IX,DE":DI=DI+1:RETURN
1820 IF N1=&HDD AND N2=&H29 THEN PRINT T
     AB(12);"ADD IX,IX":DI=DI+1:RETURN
1830 IF N1=&HDD AND N2=&H39 THEN PRINT T
     AB(12);"ADD IX,SP":DI=DI+1:RETURN
1840 IF N1=&HFD AND N2=&H9 THEN PRINT TA
     B(12);"ADD IY,BC":DI=DI+1:RETURN
1850 IF N1=&HFD AND N2=&H19 THEN PRINT T
     AB(12);"ADD IY,DE":DI=DI+1:RETURN
1860 IF N1=&HFD AND N2=&H29 THEN PRINT T
     AB(12);"ADD IY,IY":DI=DI+1:RETURN
1870 IF N1=&HFD AND N2=&H39 THEN PRINT T
     AB(12);"ADD IY,SP":DI=DI+1:RETURN
1880 IF N1=&HDD AND N2=&H2B THEN PRINT T
     AB(12);"DEC IX":DI=DI+1:RETURN
1890 IF N1=&HFD AND N2=&H2B THEN PRINT T
     AB(12);"DEC IY":DI=DI+1:RETURN
1900 IF N1=&HDD AND N2=&HE3 THEN PRINT T
     AB(12);"EX (SP),IX":DI=DI+1:RETURN
1910 IF N1=&HFD AND N2=&HE3 THEN PRINT T
     AB(12);"EX (SP),IY":DI=DI+1:RETURN
1920 IF N1=&HDD AND N2=&H34 THEN PRINT T
     AB(12);"INC (IX+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
1930 IF N1=&HFD AND N2=&H34 THEN PRINT T
```

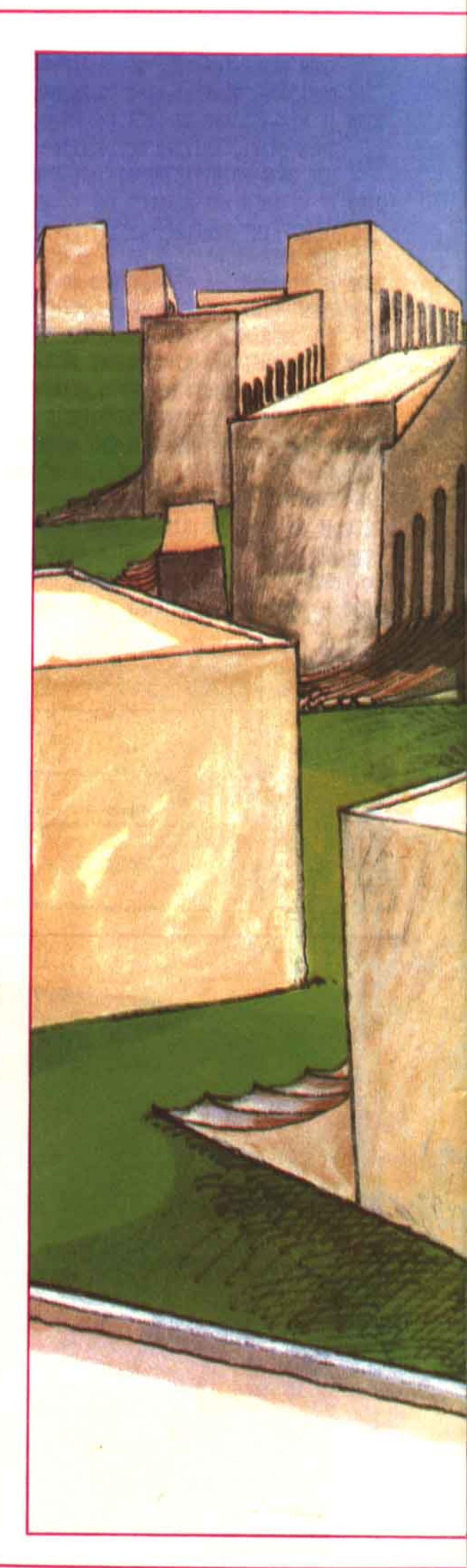
```
AB(12);"INC (IY+";:GOSUB 3030:PRINT
")":DI=DI+2:RETURN
1940 IF N1=&HDD AND N2=&H23 THEN PRINT T
AB(12);"INC IX":DI=DI+1:RETURN
1950 IF N1=&HFD AND N2=&H23 THEN PRINT T
AB(12);"INC IY":DI=DI+1:RETURN
1960 IF N1=&HDD AND N2=&HE9 THEN PRINT T
AB(12);"JP (IX)":DI=DI+1:RETURN
1970 IF N1=&HFD AND N2=&HE9 THEN PRINT T
AB(12);"JP (IY)":DI=DI+1:RETURN
1980 IF N1=&HDD AND N2=&H22 THEN PRINT T
AB(12);"LD (";:GOSUB 3010:PRINT"),I
X":DI=DI+3:RETURN
1990 IF N1=&HFD AND N2=&H22 THEN PRINT T
AB(12);"LD (";:GOSUB 3010:PRINT"),I
Y":DI=DI+3:RETURN
2000 IF N1=&HDD AND N2=&H77 THEN PRINT T
AB(12);"LD (IX+";:GOSUB 3030:PRINT"
),A":DI=DI+2:RETURN
2010 IF N1=&HDD AND N2=&H70 THEN PRINT T
AB(12);"LD (IX+";:GOSUB 3030:PRINT"
),B":DI=DI+2:RETURN
2020 IF N1=&HDD AND N2=&H71 THEN PRINT T
AB(12);"LD (IX+";:GOSUB 3030:PRINT"
),C":DI=DI+2:RETURN
2030 IF N1=&HDD AND N2=&H72 THEN PRINT T
AB(12);"LD (IX+";:GOSUB 3030:PRINT"
),D":DI=DI+2:RETURN
2040 IF N1=&HDD AND N2=&H36 THEN PRINT T
AB(12);"LD (IX+";:GOSUB 3030:PRINT"
),";:DI=DI+1:GOSUB 3030:PRINT:DI=DI
+2:RETURN
2050 IF N1=&HDD AND N2=&H73 THEN PRINT T
AB(12);"LD (IX+";:GOSUB 3030:PRINT"
),E":DI=DI+2:RETURN
2060 IF N1=&HDD AND N2=&H74 THEN PRINT T
AB(12);"LD (IX+";:GOSUB 3030:PRINT"
),H":DI=DI+2:RETURN
2070 IF N1=&HDD AND N2=&H75 THEN PRINT T
AB(12);"LD (IX+";:GOSUB 3030:PRINT"
),L":DI=DI+2:RETURN
2080 IF N1=&HFD AND N2=&H77 THEN PRINT T
AB(12);"LD (IY+";:GOSUB 3030:PRINT"
),A":DI=DI+2:RETURN
2090 IF N1=&HFD AND N2=&H70 THEN PRINT T
AB(12);"LD (IY+";:GOSUB 3030:PRINT"
),B":DI=DI+2:RETURN
2100 IF N1=&HFD AND N2=&H71 THEN PRINT T
AB(12);"LD (IY+";:GOSUB 3030:PRINT"
),C":DI=DI+2:RETURN
2110 IF N1=&HFD AND N2=&H72 THEN PRINT T
AB(12);"LD (IY+";:GOSUB 3030:PRINT"
),D":DI=DI+2:RETURN
2120 IF N1=&HFD AND N2=&H36 THEN PRINT T
AB(12);"LD (IY+";:GOSUB 3030:PRINT"
),";:DI=DI+1:GOSUB 3030:PRINT:DI=DI
+2:RETURN
2130 IF N1=&HFD AND N2=&H73 THEN PRINT T
AB(12);"LD (IY+";:GOSUB 3030:PRINT"
),E":DI=DI+2:RETURN
2140 IF N1=&HFD AND N2=&H74 THEN PRINT T
AB(12);"LD (IY+";:GOSUB 3030:PRINT"
),H":DI=DI+2:RETURN
2150 IF N1=&HFD AND N2=&H75 THEN PRINT T
AB(12);"LD (IY+";:GOSUB 3030:PRINT"
),L":DI=DI+2:RETURN
2160 IF N1=&HDD AND N2=&H7E THEN PRINT T
AB(12);"LD A,(IX+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2170 IF N1=&HFD AND N2=&H7E THEN PRINT T
AB(12);"LD A,(IY+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2180 IF N1=&HDD AND N2=&H46 THEN PRINT T
AB(12);"LD B,(IX+";:GOSUB 3030:PIRN
T")":DI=DI+2:RETURN
2190 IF N1=&HFD AND N2=&H46 THEN PRINT T
AB(12);"LD B,(IY+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2200 IF N1=&HDD AND N2=&H4E THEN PRINT T
AB(12);"LD C,(IX+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2210 IF N1=&HFD AND N2=&H4E THEN PRINT T
AB(12);"LD C,(IY+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2220 IF N1=&HDD AND N2=&H56 THEN PRINT T
AB(12);"LD D,(IX+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2230 IF N1=&HFD AND N2=&H56 THEN PRINT T
AB(12);"LD D,(IY+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2240 IF N1=&HDD AND N2=&H5E THEN PRINT T
AB(12);"LD E,(IX+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2250 IF N1=&HFD AND N2=&H5E THEN PRINT T
AB(12);"LD E,(IY+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2260 IF N1=&HDD AND N2=&H66 THEN PRINT T
AB(12);"LD H,(IX+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2270 IF N1=&HFD AND N2=&H66 THEN PRINT T
AB(12);"LD H,(IY+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2280 IF N1=&HDD AND N2=&H2A THEN PRINT T
AB(12);"LD IX,(";:GOSUB 3010:PRINT"
)":DI=DI+3:RETURN
2290 IF N1=&HFD AND N2=&H2A THEN PRINT T
AB(12);"LD IY,(";:GOSUB 3010:PRINT"
)":DI=DI+3:RETURN
2300 IF N1=&HDD AND N2=&H21 THEN PRINT T
AB(12);"LD IX,";:GOSUB 3010:PRINT:D
I=DI+3:RETURN
2310 IF N1=&HFD AND N2=&H21 THEN PRINT T
AB(12);"LD IY,";:GOSUB 3010:PRINT:D
I=DI+3:RETURN
2320 IF N1=&HDD AND N2=&H6E THEN PRINT T
AB(12);"LD L,(IX+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2330 IF N1=&HFD AND N2=&H6E THEN PRINT T
AB(12);"LD L,(IY+";:GOSUB 3030:PRIN
T")":DI=DI+2:RETURN
2340 IF N1=&HDD AND N2=&HF9 THEN PRINT T
AB(12);"LD SP,IX":DI=DI+1:RETURN
2350 IF N1=&HFD AND N2=&HF9 THEN PRINT T
AB(12);"LD SP,IY":DI=DI+1:RETURN
2360 IF N1=&HDD AND N2=&HB6 THEN PRINT T
AB(12);"OR (IX+";:GOSUB 3030:PRINT"
)":DI=DI+2:RETURN
2370 IF N1=&HFD AND N2=&HB6 THEN PRINT T
AB(12);"OR (IY+";:GOSUB 3030:PRINT"
)":DI=DI+2:RETURN
2380 IF N1=&HDD AND N2=&HE1 THEN PRINT T
AB(12);"POP IX":DI=DI+1:RETURN
2390 IF N1=&HFD AND N2=&HE1 THEN PRINT T
AB(12);"POP IY":DI=DI+1:RETURN
2400 IF N1=&HDD AND N2=&HE5 THEN PRINT T
AB(12);"PUSH IX":DI=DI+1:RETURN
2410 IF N1=&HFD AND N2=&HE5 THEN PRINT T
AB(12);"PUSH IY":DI=DI+1:RETURN
2420 IF N1=&HDD AND N2=&HCB AND N4=&H86
THEN PRINT TAB(12);"RES 0,(IX+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2430 IF N1=&HFD AND N2=&HCB AND N4=&H86
THEN PRINT TAB(12);"RES 0,(IY+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2440 IF N1=&HDD AND N2=&HCB AND N4=&H8E
THEN PRINT TAB(12);"RES 1,(IX+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2450 IF N1=&HFD AND N2=&HCB AND N4=&H8E
THEN PRINT TAB(12);"RES 1,(IY+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2460 IF N1=&HDD AND N2=&HCB AND N4=&H96
THEN PRINT TAB(12);"RES 2,(IX+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2470 IF N1=&HFD AND N2=&HCB AND N4=&H96
THEN PRINT TAB(12);"RES 2,(IY+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2480 IF N1=&HDD AND N2=&HCB AND N4=&H9E
THEN PRINT TAB(12);"RES 3,(IX+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2490 IF N1=&HFD AND N2=&HCB AND N4=&H9E
THEN PRINT TAB(12);"RES 3,(IY+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2500 IF N1=&HDD AND N2=&HCB AND N4=&HA6
THEN PRINT TAB(12);"RES 4,(IX+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2510 IF N1=&HFD AND N2=&HCB AND N4=&HA6
THEN PRINT TAB(12);"RES 4,(IY+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2520 IF N1=&HDD AND N2=&HCB AND N4=&HAE
THEN PRINT TAB(12);"RES 5,(IX+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
2530 IF N1=&HFD AND N2=&HCB AND N4=&HAE
THEN PRINT TAB(12);"RES 5,(IY+";:GO
SUB 3030:PRINT")":DI=DI+3:RETURN
```

```
2540 IF N1=&HFD AND N2=&HCB AND N4=&H6 T
     HEN PRINT TAB(12);"RLC (IY+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2550 IF N1=&HDD AND N2=&HCB AND N4=&HB6
     THEN PRINT TAB(12);"RES 6,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2560 IF N1=&HFD AND N2=&HCB AND N4=&HB6
     THEN PRINT TAB(12);"RES 6,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2570 IF N1=&HDD AND N2=&HCB AND N4=&HBE
     THEN PRINT TAB(12);"RES 7,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2580 IF N1=&HFD AND N2=&HCB AND N4=&HBE
     THEN PRINT TAB(12);"RES 7,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2590 IF N1=&HDD AND N2=&HCB AND N4=&H16
     THEN PRINT TAB(12);"RL (IX+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2600 IF N1=&HFD AND N2=&HCB AND N4=&H16
     THEN PRINT TAB(12);"RL (IY+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2610 IF N1=&HDD AND N2=&HCB AND N4=&H6 T
     HEN PRINT TAB(12);"RLC (IX+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2620 IF N1=&HFD AND N2=&HCB AND N4=&H6 T
     HEN PRINT TAB(12);"RLC (IY+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2630 IF N1=&HDD AND N2=&HCB AND N4=&H1E
     THEN PRINT TAB(12);"RR (IX+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2640 IF N1=&HFD AND N2=&HCB AND N4=&H1E
     THEN PRINT TAB(12);"RR (IY+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2650 IF N1=&HDD AND N2=&HCB AND N4=&HE T
     HEN PRINT TAB(12);"RRC (IX+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2660 IF N1=&HFD AND N2=&HCB AND N4=&HE T
     HEN PRINT TAB(12);"RRC (IY+";:GOSUB
      3030:PRINT")":DI=DI+3:RETURN
2670 IF N1=&HDD AND N2=&H9E THEN PRINT T
     AB(12);"SBC A,(IX+";:GOSUB 3030:PRI
     NT")":DI=DI+2:RETURN
2680 IF N1=&HFD AND N2=&H9E THEN PRINT T
     AB(12);"SBC A,(IY+";:GOSUB 3030:PRI
     NT")":DI=DI+2:RETURN
2690 IF N1=&HDD AND N2=&HCB AND N4=&HC6
     THEN PRINT TAB(12);"SET 0,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2700 IF N1=&HFD AND N2=&HCB AND N4=&HC6
     THEN PRINT TAB(12);"SET 0,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2710 IF N1=&HDD AND N2=&HCB AND N4=&HCE
     THEN PRINT TAB(12);"SET 1,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2720 IF N1=&HFD AND N2=&HCB AND N4=&HCE
     THEN PRINT TAB(12);"SET 1,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2730 IF N1=&HDD AND N2=&HCB AND N4=&HD6
     THEN PRINT TAB(12);"SET 2,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2740 IF N1=&HFD AND N2=&HCB AND N4=&HD6
     THEN PRINT TAB(12);"SET 2,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2750 IF N1=&HDD AND N2=&HCB AND N4=&HDE
     THEN PRINT TAB(12);"SET 3,(IX+";:GO
```



```
     SUB 3030:PRINT")":DI=DI+3:RETURN
2760 IF N1=&HFD AND N2=&HCB AND N4=&HDE
     THEN PRINT TAB(12);"SET 3,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2770 IF N1=&HDD AND N2=&HCB AND N4=&HE6
     THEN PRINT TAB(12);"SET 4,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2780 IF N1=&HFD AND N2=&HCB AND N4=&HE6
     THEN PRINT TAB(12);"SET 4,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2790 IF N1=&HDD AND N2=&HCB AND N4=&HEE
     THEN PRINT TAB(12);"SET 5,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2800 IF N1=&HFD AND N2=&HCB AND N4=&HEE
     THEN PRINT TAB(12);"SET 5,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2810 IF N1=&HDD AND N2=&HCB AND N4=&HF6
     THEN PRINT TAB(12);"SET 6,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2820 IF N1=&HFD AND N2=&HCB AND N4=&HF6
     THEN PRINT TAB(12);"SET 6,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2830 IF N1=&HDD AND N2=&HCB AND N4=&HFE
     THEN PRINT TAB(12);"SET 7,(IX+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2840 IF N1=&HFD AND N2=&HCB AND N4=&HFE
     THEN PRINT TAB(12);"SET 7,(IY+";:GO
     SUB 3030:PRINT")":DI=DI+3:RETURN
2850 IF N1=&HDD AND N2=&HCB AND N4=&H26
     THEN PRINT TAB(12);"SLA (IX+";:GOSU
     B 3030:PRINT")":DI=DI+3:RETURN
2860 IF N1=&HFD AND N2=&HCB AND N4=&H26
     THEN PRINT TAB(12);"SLA (IY+";:GOSU
     B 3030:PRINT")":DI=DI+3:RETURN
2870 IF N1=&HDD AND N2=&HCB AND N4=&H2E
     THEN PRINT TAB(12);"SRA (IX+";:GOSU
     B 3030:PRINT")":DI=DI+3:RETURN
2880 IF N1=&HFD AND N2=&HCB AND N4=&H2E
     THEN PRINT TAB(12);"SRA (IY+";:GOSU
     B 3030:PRINT")":DI=DI+3:RETURN
2890 IF N1=&HDD AND N2=&HCB AND N4=&H3E
     THEN PRINT TAB(12);"SRL (IX+";:GOSU
     B 3030:PRINT")":DI=DI+3:RETURN
2900 IF N1=&HFD AND N2=&HCB AND N4=&H3E
     THEN PRINT TAB(12);"SRL (IY+";:GOSU
     B 3030:PRINT")":DI=DI+3:RETURN
2910 IF N1=&HDD AND N2=&H96 THEN PRINT T
     AB(12);"SUB (IX+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
2920 IF N1=&HFD AND N2=&H96 THEN PRINT T
     AB(12);"SUB (IY+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
2930 IF N1=&HDD AND N2=&HAE THEN PRINT T
     AB(12);"XOR (IX+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
2940 IF N1=&HFD AND N2=&HAE THEN PRINT T
     AB(12);"XOR (IY+";:GOSUB 3030:PRINT
     ")":DI=DI+2:RETURN
2950 RETURN
2960 FOR J=0 TO I
2970 READ I$
2980 NEXT J
2990 PRINT TAB(12);I$;
3000 RETURN
3010 IF PEEK(DI+2)<=&HF THEN PRINT"0";
3020 PRINT HEX$(PEEK(DI+2));
3030 IF PEEK(DI+1)<=&HF THEN PRINT"0";
3040 PRINT HEX$(PEEK(DI+1));
3050 RETURN
```

Código máquina MSX



Como comentábamos en el número anterior, vamos a continuar centrándonos en el análisis de problemas sencillos para ver cómo se llega, desbozándolos poco a poco, a su solución.

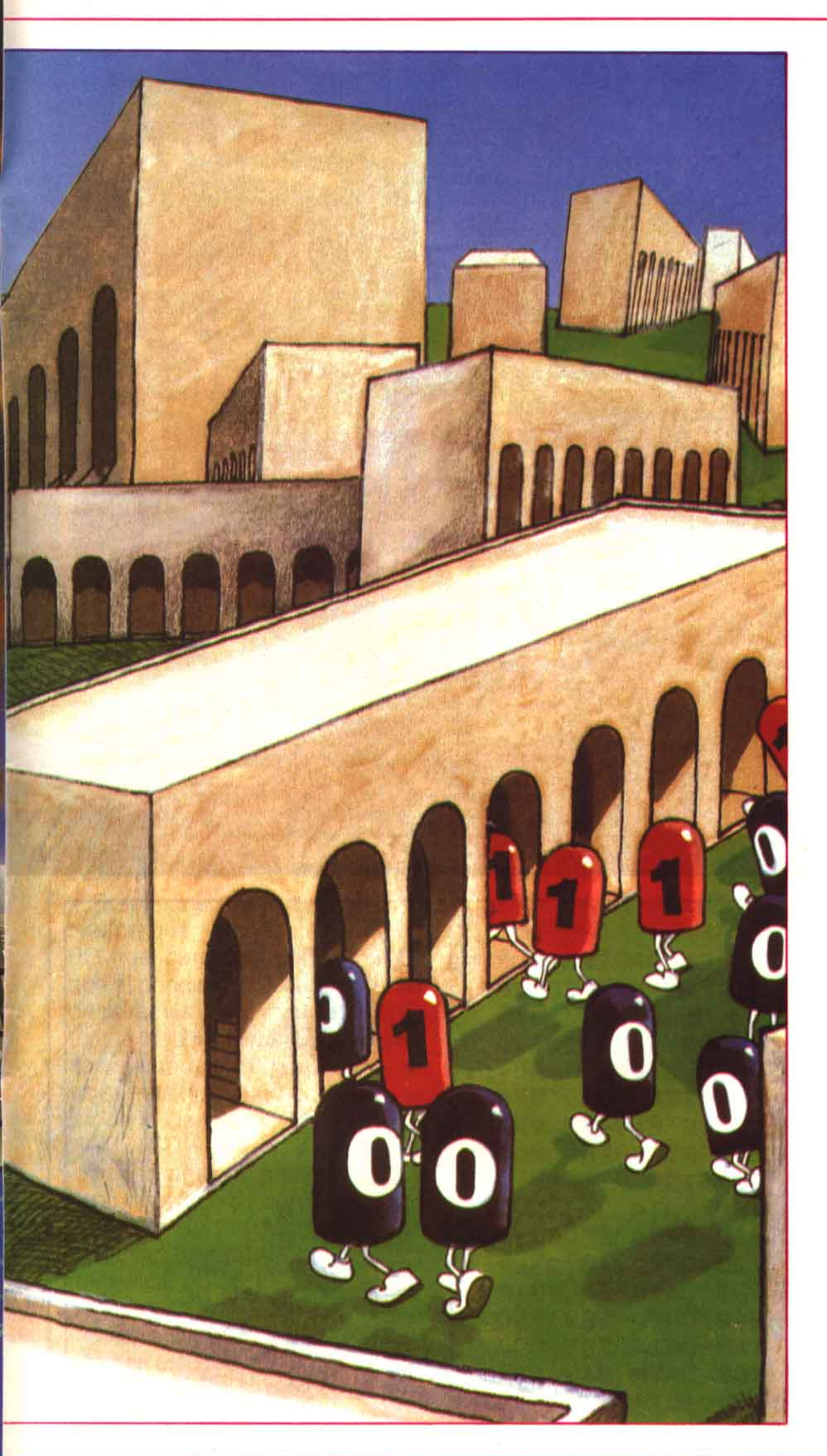
En esta ocasión el problema que os proponemos es el diseño de una rutina que multiplique dos números. Para no complicar en exceso la cuestión, vamos a limitarnos a multiplicar dos números enteros (esto es, sin decimales) para obtener otro número entero.

El análisis

Dado que desde nuestra más tierna infancia aprendemos las operaciones matemáticas básicas, llega un momento en que las realizamos de forma automática. Por ello, este problema reviste una dificultad algo especial, ya que debemos descubrir cómo hacemos algo que ya hemos incorporado a nuestras costumbres rutinarias.

Veamos un ejemplo: 350 x 128. Normalmente, para multiplicar estos dos números, colocaríamos 350 arriba y 128 abajo, y procederíamos así: ocho por cero, cero; ocho por cinco cuarenta (cero, y me llevo cuatro); ocho por tres veinticuatro y cuatro veintiocho. De este modo obtenemos una primera línea con el número 2800.

A continuación procedemos con el dos, pero hay que destacar que los resultados comenzamos a escribirlos UNA POSICION HACIA LA IZQUIERDA. Así, haríamos: dos por cero, cero (y colocamos el cero DEBAJO DEL SEGUNDO CERO POR LA DERECHA); dos por cinco diez y me llevo una (y colocamos el cero DEBAJO DEL OCHO); dos por tres seis y una siete (y colocamos el siete DEBAJO DEL DOS). Dado que, una vez terminada la



multiplicación, vamos a sumar los tres parciales obtenidos, pero estado el segundo desplazado un lugar a la izquierda y el segundo dos lugares a la izquierda, lo que en realidad hacemos es la siguiente operación:

(350 x 8)+(350 x 20)+(350 x 100)

Podéis comprobarlo realizando esta operación y comparando con el resultado obtenido mediante la multiplicación habitual.

Hemos de realizar una observación, aunque muchos ya se habrán dado cuenta de la importancia de este detalle: si un número decimal lo desplazamos un lugar a la izquierda, rellenando por la derecha con un cero, obtenemos el mismo número multiplicado por diez. Por ejemplo, 1452, al desplazarlo a la izquierda y colocar un cero a la derecha, se convierte en 14520 = 1452 x 10.

Esto mismo, trasladado al mundo de los números binarios, significa que al rotar un número a la izquierda y rellenar por la derecha con ceros lo multiplicamos por dos. Por ejemplo, el número binario 00101100 = 44, al rotarlo a la derecha se convierte en 01011000 = 88. La importancia de esta observación reside en que el Z80 posee la instrucción genérica RL r, donde r es un registro de 8 *bits* (A, B, C, D, E, H, L) y RL es la abreviatura de *Rotable Left* (Rotar a la izquierda), que realiza justamente esa operación: rotar el contenido del registro a la izquierda. Aunque por la derecha no rellenamos con ceros, sino con el contenido del *CARRY,* veremos que en este caso no nos importa realmente con qué rellenemos.

Hay que tener en cuenta que, al trabajar con el sistema de numeración binario, la multiplicación se reduce a dos casos de suma: si en el multiplicador tenemos un uno,

sumamos el multiplicando tal cual, y si tenemos un cero no lo sumamos. Por ejemplo, 0100 x 0110:

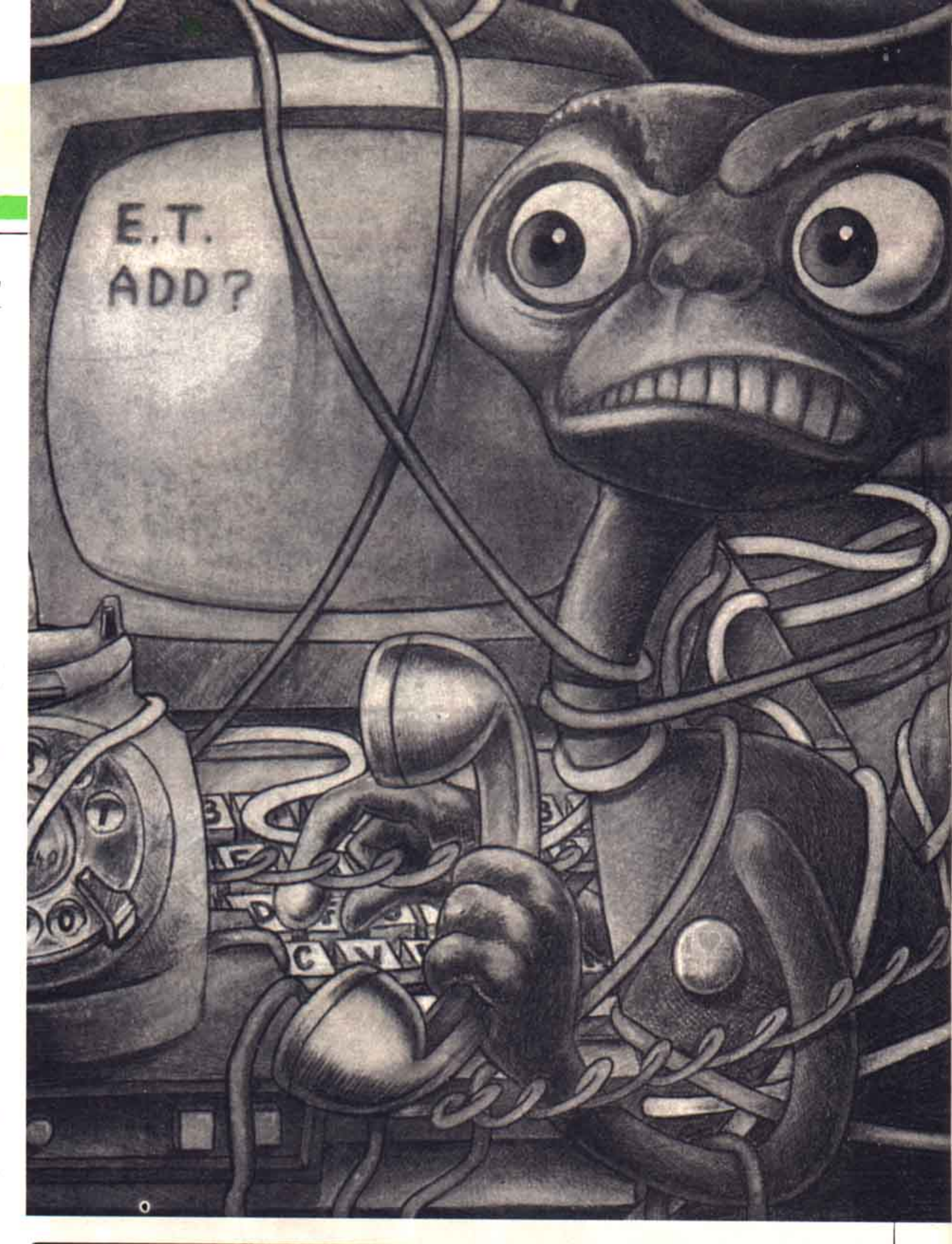
```
       0100
     X 0110
   ---------
       0000
      0100
     0100
    0000     +
   ---------
    0011000
```

Por tanto, nuestra rutina se va a limitar a ir rotando el multiplicador a la izquierda y examinando el *CARRY*. Si el *CARRY* está activado es que hemos topado con un uno, y sumamos el multiplicando (registro DE) al resultado (registro HL). Como veis, vamos sumando cada resultado parcial según lo obtenemos, ya que si no sería necesario ir guardando en memoria todos los parciales para luego sumarlos.

Tal vez os extrañe la instrucción ADD HL,HL. A primera vista no tiene sentido sumar el resultado a sí mismo, pero hay que tener en cuenta que sumar un número a sí mismo es multiplicarlo por dos, y como ya hemos comentado, en numeración binaria multiplicar por dos es lo mismo que rotar a la izquierda una posición. Por tanto, la función de la instrucción ADD HL,HL es rotar a la izquierda el resultado. Esto se debe a que, para simplificar y hacer más rápida la rutina, hemos invertido el proceso normal de la multiplicación, y en lugar de comenzar a multiplicar considerando el multiplicador de derecha a izquierda, lo hacemos de izquierda a derecha.

La función USR

Hay algunas peculiaridades de esta función que merece la pena comentar con detalle. Esta función, como ya sabréis, nos permite




```
10 CLEAR 200,&HF1FF
20 FOR D=0 TO 102
30 READ A$
40 POKE &HF200+D,VAL("&H"+A$)
50 S=S+VAL("&H"+A$)
60 NEXT D
70 IF S=9474 THEN 640
80 PRINT"ERROR EN LOS DATAS"
90 STOP
100 DATA 3A,63,F6:    REM LD A,(TIPO)
110 DATA FE,02:       REM CP 02H
120 DATA 20,25:       REM JR NZ,ERROR1
130 DATA 2A,F8,F7:    REM LD HL,(LOC)
140 DATA 5E:          REM LD E,(HL)
150 DATA 23:          REM INC HL
160 DATA 56:          REM LD D,(HL)
```

ejecutar rutinas en código máquina. Además, podemos transmitirle a la rutina un parámetro (el que colocamos entre paréntesis al llamar a la función), y ésta nos puede devolver también un valor. En ambos casos se tratará de números enteros.

¿Y qué ocurre si queremos enviar a la rutina más de un parámetro? Pues muy sencillo: basta con preparar una matriz con los parámetros a enviar y proporcionarle a nuestra subrutina la dirección de comienzo de la matriz. Dicha dirección de comienzo se obtiene aplicando la función VARPTR al primer elemento de la matriz (el elemento con subíndice cero).

En el caso concreto de nuestro programa necesitamos enviar a la rutina los dos números enteros a multiplicar. Para ello hemos preparado la matriz entera M% (línea 640), en la cual introducimos los dos números a multiplicar (líneas 670 y 680). Luego averiguamos la dirección del elemento M%(0) (línea 690), que es el valor que le proporcionamos a nuestra rutina cuando la llamamos (línea 810).

Examinando el listado ensamblador que encontraréis en los REM del cargador BASIC, vemos que al principio verificamos el tipo de parámetro que recibe la rutina. Como las direcciones de memoria son siempre números enteros, si no se trata de un número entero se produce el mensaje de error «PARAMETRO ILEGAL». A continuación cargamos el parámetro (que es la dirección de comienzo de la matriz) en HL, y lo utilizamos como puntero para cargar los números a multiplicar en DE y en el falso par de registros AC. Luego inicializamos el registro B con 16 (10H). Utilizamos este registro como contador del número de desplazamientos a realizar, o lo que es lo mismo,

```
170 DATA 23:                REM INC HL
180 DATA 4E:                REM LD C,(HL)
190 DATA 23:                REM INC HL
200 DATA 7E:                REM LD A,(HL)
210 DATA 06,10:             REM LD B,10H
220 DATA 21,00,00:          REM LD HL,0000H
230                         REM LAZO1
240 DATA 29:                REM ADD HL,HL
250 DATA 38,1A:             REM JR C,ERROR2
260 DATA CB,11:             REM RL C
270 DATA 17:                REM RLA
280 DATA 30,03:             REM JR NC,SIGUE
290 DATA 19:                REM ADD HL,DE
300 DATA 38,12:             REM JR C,ERROR2
310                         REM SIGUE
320 DATA 10,F3:             REM DJNZ LAZO1
330 DATA 3E,02:             REM LD A,02H
340 DATA 32,63,F6:          REM LD (TIPO),A
350 DATA 22,F8,F7:          REM LD (LOC),HL
360 DATA C9:                REM RET
370                         REM ERROR1
380 DATA 21,44,F2:          REM LD HL,TEXTO1
390 DATA CD,3A,F2:          REM CALL PRINT
400 DATA C9:                REM RET
410                         REM ERROR2
420 DATA 21,56,F2:          REM LD HL,TEXTO2
430 DATA CD,3A,F2:          REM CALL PRINT
440 DATA C9:                REM RET
450                         REM PRINT
460 DATA 7E:                REM LD A,(HL)
470 DATA CD,A2,00:          REM CALL 00A2H
480 DATA FE,0A:             REM CP 0AH
490 DATA C8:                REM RET Z
500 DATA 23:                REM INC HL
510 DATA 18,F6:             REM JR PRINT
520                         REM TEXTO1
530 DATA 50,41,52,41:REM DB 'PARA'
540 DATA 4D,45,54,52:REM DB 'METR'
550 DATA 4F,20,49,4C:REM DB 'O IL'
560 DATA 45,47,41,4C:REM DB 'EGAL'
570 DATA 0D,0A:             REM DB 0DH,0AH
580                         REM TEXTO2
590 DATA 53,4F,42,52:REM DB 'SOBR'
600 DATA 45,50,41,53:REM DB 'EPAS'
610 DATA 41,4D,49,45:REM DB 'AMIE'
620 DATA 4E,54,4F:    REM DB 'NTO'
630 DATA 0D,0A:             REM DB 0DH,0AH
640 DIM M%(2)
650 DEF FN C1(X)=-(X<32768!)*X-(X>=327
```

del número de bits del multiplicador.

El resto de la rutina ya lo hemos comentado, excepto el final. Una vez completada la multiplicación, y si no se ha producido ningún error de sobrepasamiento (error que se produce si el producto obtenido necesita más de dieciséis bits), se coloca el resultado en las direcciones F7F8H y F7F9H (línea 350) e informamos al BASIC que le devolvemos un valor entero introduciendo 02H en la dirección F663H (líneas 330 y 340). El resto del programa lo forman las rutinas de error. Ambas cargan en HL la dirección de comienzo del correspondiente mensaje de error y llaman a la rutina de impresión PRINT.

```
    68!)*(X-65536!)
660 DEF FN C2(X)=-(X>=0)*X-(X<0)*(6553
    6!+X)
670 M%(0)=FN C1(10000)
680 M%(1)=FN C1(4)
690 A%=VARPTR(M%(0))
700 DEF USR=&HF200
710 CLS
720 PRINT"EJEMPLO"
730 PRINT
740 PRINT"PRODUCTO DE 10000 Y 4"
750 PRINT
760 PRINT"POR BASIC"
770 PRINT"10000 X 4 ="10000*4
780 PRINT
790 PRINT"POR CODIGO MAQUINA"
800 PRINT"10000 X 4 =";
810 PRINT FN C2(USR(A%))
820 PRINT
830 END
```

compro, vendo, cambio...

• VENDO ampliación de memoria MSX de 16K, semi-nueva de Sony por 5.000 ptas. Llamar a José al Tel.: (91) 255 05 56 de 22 a 23 horas.

• VENDO dos Hit Bit de Sony. Uno de 16K y otro de 64K. Unidad, diskettes, manuales, revistas, cables y programas. Escribir a José Bua González. Apartado 78, Boiro (La Coruña) 15931 o llamar al Tel.: (981) 24 42 74.

• SE CAMBIAN toda clase de programas en cinta. Escribir a Santiago Alvarez Gómez. José Antonio, 55. Chantada (Lugo) 27500.

• INTERCAMBIO programas de juegos, educativos, utilidades y gestión. Escribir a Carmelo González García. Industrias, 34, 5-C. Valladolid 47005, o llamar al Tel.: (983) 29 45 92.

• VENDO ordenador Philips VG-8010, con varios programas y 25 revistas, todo por 25.000 ptas. Llamar a partir de las 18h. al Tel.: (967) 25 00 52 o escribir a Tomás García. Mayor, 1. Pozo-Cañada (Albacete).

• INTERCAMBIO programas de MSX. Escribir a Javier Mesanza. Perú, 10, 3-I. Vitoria (Alava) o llamar al Tel.: (964) 24 44 70.

• INTERCAMBIO programas de MSX. Escribir a Leoncio Muñoz Sánchez. Polígono Canyelles, bloque C-11, bajo. 08033 Barcelona.

• INTERCAMBIO programas. Escribir a José R. García. Carretera, 27, 18811 Zújar (Granada) o llamar al Tel.: 71 61 61.

• COMPRO programas de RTTY-CW y lo relacionado con radioaficionados para MSX. Intercambio toda clase de programas. Escribir a Alejandro García. Avda. Hilario Ruis, 46. 15621 Cabañas (La Coruña).

• INTERCAMBIO programas e información sobre lenguaje Código Máquina para SVI-328. Escribir a Raúl de Frutos Tena. La Gavilla, 7, bajo B. 470145 Valladolid.

• INTERCAMBIO programas. Escribir a Fernando Alvarez Reguant. Avda. Santa Bárbara, 41. 08260 Suria (Barcelona).

• SI TE INTERESA el intercambio de programas, revistas, accesorios, etc. Escribe a Fernando Zúñiga Torrecilla. Barrio de Nuestra Señora 32, 2-D. 50007 Zaragoza.

ANUNCIOS GRATUITOS

Todos los anuncios (compras, ventas, cambios o comunicaciones de clubs de usuarios, etc.) que van en esta sección, deben tener un máximo de cuarenta palabras. Con el fin de facilitar la transcripción de los anuncios hemos recuadrado cuarenta espacios para que en cada uno vaya una palabra. Después, recortar y mandar a:

MSX MAGAZINE
ANUNCIOS GRATUITOS
C/ Bravo Murillo, 377, 5.° A
28020 MADRID

Rincón del lector

¿PASCAL PARA MSX?

Este año voy a empezar a estudiar Informática en la Universidad de Murcia, pero no puedo asistir habitualmente a las clases por razones que no vienen al caso. El problema es que poseo un Philips MSX, pero resulta que este año se va a trabajar con PASCAL, mientras que el año pasado se dio BASIC.

Mi pregunta es: ¿está el MSX capacitado para leer PASCAL o existe algún medio para hacer que mi ordenador lo pueda interpretar?

Alfredo Roldán Mazón
Murcia

Los MSX pueden trabajar con los lenguajes de programación más diversos, siempre y cuando, claro está, que poseamos los compiladores apropiados para el caso. Si tienes un compilador PASCAL, no tendrás problema alguno para seguir las clases desde casa y es más, sacarás más provecho de la programación en casa que en la Universidad, donde la masificación de las aulas presenta un problema muy importante.

SONY comercializa un compilador PASCAL para MSX. En cualquier tienda especializada lo encontrarás. Aunque como último recurso, siempre puedes recurrir a la casa. Su dirección es:
Sony España, S.A.
Sabino de Arana, 42-44
08028 Barcelona

DUDAS DIVERSAS

¿Cuánto tiempo puede estar un monitor en color funcionando de forma continua, sin que éste se resienta? Esta pregunta viene al caso debido a que utilizo mucho el ordenador? ¿El chip de sonido del Commodore 64 es mejor que el de los MSX?

Jorge Sansón-Chirinos
Tenerife

Actualmente, tanto los ordenadores, como los periféricos están diseñados para soportar las condiciones más extremas de uso y abuso a que son sometidos. Por esta razón, no debes preocuparte ni por tu ordenador ni por el monitor, ya que mientras los tengas conectados en un sitio donde circule aire, no habrá problemas. Particularmente, en nuestra redacción, hemos llegados a tener conectado un monitor y un ordenador durante más de 16 horas sin observar el más mínimo fallo.

Referente al chip de sonido, el que posee el Commodore 64 es mejor que el de los MSX. Es más completo, sin embargo, es mucho más difícil de manejar y programar, ya que todas las instrucciones que controlan el chip hay que darlas mediante POKEs. Esta es la ventaja que tiene MSX sobre el C-64. Mientras, en aquél, las instrucciones de sonido son PLAY, SOUND, etc., en el C-64 hay que introducir infinidad de POKEs (con el consiguiente riesgo de equivocación) para lograr algún ruido que se aproxime a música.

De cualquier manera, sabiendo programar el chip de sonido del C-64, hay que decir ¡CHAPEAU!

ELECCION DE ORDENADOR

Me gustaría que me informasen de los precios de los ordenadores MSX (64K) que existen en el mercado, ya que estoy interesado en comprar uno y no sé cuál elegir. También, me gustaría que me orientasen sobre los ordenadores más vendidos, los mejores, etc.

Francisco Llerena Sánchez
Badajoz

Es difícil darte una idea concreta sobre los mejores ordenadores MSX del mercado, así como los más vendidos, ya que, por ejemplo, si tomamos esta última característica, hay que tener en cuenta la diversidad de ordenadores existentes dentro de una misma marca.

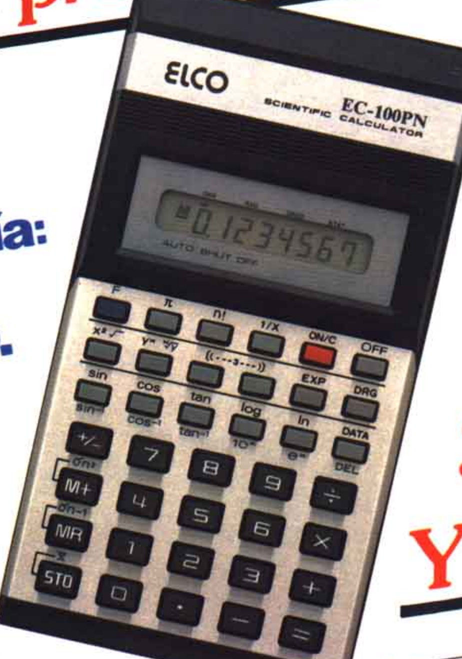
Esto haría inclinar la balanza hacia aquellos fabricantes con más diversidad de ordenadores dentro de la norma MSX, como sería el caso de Sony o Philips, sin ir más lejos. Cada uno de ellos tiene más de cuatro ordenadores, todos compatibles MSX y con unas cualidades muy particulares, que cumplen a la perfección todos los requisitos del estándar. Si fuera por precio, estamos en las mismas. Las versiones básicas, con la actual bajada de precios, se han puesto al alcance de todo el mundo.

Ahora bien, si lo que deseas son prestaciones, habrá que gastarse algo más. Ten en cuenta que, si compras algún ordenador de la I generación, tendrás que aportar algo más de dinero si lo deseas con unidad de disco.

De cualquier manera, inclínate hacia los de la II generación si quieres buenas prestaciones y un ordenador para algo más que jugar a los marcianos.

ELCO calculadoras para estudiantes:

Por el precio de una calculadora sencilla



Garantía: UN AÑO.

2.990 ptas.

E C - 1 0 0 P N
LA CIENTIFICA ECONOMICA

Pantalla en LCD con 8 dígitos (5+2).
Funciones trigonométricas, logarítmicas, exponenciales y sus inversas. Grados centígrados, sexagesimales y radianes.
Factoriales, radicales, funciones estadísticas (media, varianza desviación típica).
AOS (sistema operativo Algebráico).
Apagado automático.
Alimentación con dos pilas normales. Duración aproximadamente 1 año.

31 FUNCIONES Y ESTADISTICA



E C - 3 9 0 LA LIGERA
31 Funciones con estadísticas y 8 dígitos.
Apagado automático.
3.290 ptas.



E C - 5 9 0 II
LA CIENTIFICA COMPLEJA
94 funciones y 12 dígitos.
Memoria constante.
Conversiones y cálculos en binario, hexadécimal, octal y décimal.
4.590 ptas.

E C S - 9 9 0 II
LA SOLAR
94 funciones y 12 dígitos.
Conversiones y cálculo en binario, hexadécimal, octal y décimal.
Celdas solares de alta resolución.
5.590 ptas.



E C P - 3 . 9 0 0
LA PROGRAMABLE
Admite dos programas y 45 pasos de programación en memoria constante.
Con toma de decisiones.
64 funciones científicas y 10 dígitos.
6.590 ptas.

ALVARO SOBRINO

ELCO-1 Electrónica de Consumo-1, S.A.

c/ Rufino González, 6
Telfs.: 204 76 56 y 204 05 70 - Telex 42489 ELCO E
28037 MADRID